IJCLA ISSN 0976-0962

International Journal of Computational Linguistics and Applications

Vol. 6 No. 1 Jan-Jun 2015

Editor-in-Chief
ALEXANDER GELBUKH
Instituto Politécnico Nacional, Mexico

ISSN 0976-0962

International Journal of Computational Linguistics and Applications

Vol. 6 **No. 1** **Jan-Jun 2015**

International Journal of Computational Linguistics and Applications – IJCLA (started in 2010) is a peer-reviewed international journal published twice a year, in June and December. It publishes original research papers related to computational linguistics, natural language processing, human language technologies and their applications.

Indexing: Cabell's Directory of Publishing Opportunities.

Editor-in-Chief: Alexander Gelbukh

Subscription: Rs. 2999 / US$ 199

Payments can be made by Cheques/Bank Drafts/International Money Orders drawn in the name of BAHRI PUBLICATIONS, NEW DELHI and sent to:

BAHRI PUBLICATIONS
1749A/5, 1st Floor, Gobindpuri Extension,
Kalkaji, New Delhi 110019
Telephones: 011-65810766, (0) 9811204673, (0) 9212794543
E-mails: <bahrius@vsnl.com>; <bahripublications@yahoo.com>
Website: <http://www.bahripublications.in>

Printed & Published by **BAHRI PUBLICATIONS**, New Delhi

ISSN 0976-0962

International Journal of Computational Linguistics and Applications

Vol. 6 **No. 1** **Jan-Jun 2015**

CONTENTS

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 7–9
ISSN 0976-0962, http://ijcla.bahripublications.com

Editorial

This issue of IJCLA presents papers on social linguistics, stylometry, analysis or literary texts, sentiment analysis, text summarization, automatic essay scoring, automatic speech recognition, word sense disambiguation, semantic text similarity, and text representation.

R. Cotterill et al. (UK) present a corpus of dialogs in which participants have perceived difference in social power. They show that such dialogs can be automatically classified, with above-chance precision, by the relationship between the participants of a particular dialog, which demonstrates that the corpus presents important features that reflect such relationship. The corpus will be very useful for sociolinguistic research.

E. Davoodi & **L. Kosseim** (Canada) investigate the relationship between complexity of text, i.e., its readability level, and its discourse-level properties. On the material of Simple English Wikipedia, they show that simple text contains the same discourse relations as normal, or complex, text; however, the lexical choices for the discourse markers are affected by the desirable readability level of the text.

M.-A. Boukhaled et al. (France) propose an objective interestingness measure that allows extracting meaningful syntactic patterns for computational stylistic research without any prior knowledge. They apply their approach to classic French literature texts. An important property of their measure is that it can be applied to both long and short texts; this allows its application to literature works that do not belong to any larger collection of texts.

C. Martínez et al. (Chile) present a study of emotional charge of the texts of Chilean school textbooks from first to eighth year. They automatically determine the degree of expression of the six basic emotions (anger, sadness, fear, disgust, surprise, happiness) in the text; the performance of their

automatic procedure is evaluated by human experts. They show that happiness is by far a predominant emotion expressed in the analyzed textbooks, followed by sadness and fear. They observed that each emotion is expressed with approximately the same degree in the texts of different genres, except that anger was not observed in songs.

H. Zidoum et al. (Oman) describe the use of lexical cohesion measured with help of lexical chains for extractive text summarization of Arabic documents. Arabic language is underrepresented in computational linguistics literature in general and in the literature on text summarization in particular. The authors give a detailed step-by-step account of their algorithm and compare the obtained results with human judgements.

D. Aguirre et al. (USA) present a method for automatic evaluation of text summaries written by elementary school students, with the aim of facilitating the work of schoolteachers and improve the feedback time. They show that the use of semantic similarity measures and pre-processing such as spelling correction improve the precision of their automatic grader. Their method achieves 98% of precision at 9-point grading scale, which is comparable with the agreement between human graders.

T. Nadungodage et al. (Sri Lanka and UK) show how to reduce the number of samples necessary for speaker adaptation in automatic speech recognition. The method allows building general speaker adaptation models, which outperform speaker-independent models based on the same amount of training data. The task is important in the situation when the available training corpora are too small for training the recognition system in a traditional way. The authors apply their method to Sinhala, a low-resource language spoken by the majority of population of Sri Lanka.

M. A. Sobrevilla Cabezudo & **T. A. Salgueiro Pardo** (Brazil) address the problem of word sense disambiguation for verbs in Brazilian Portuguese. Verbs are more difficult to disambiguate than nouns, and, while well studied for English, word sense disambiguation in Portuguese has received relatively little attention in literature. The authors use the Portuguese

WordNet, WordNet-Pr, as the sense inventory. Their experiments show poor performance of existing methods, which implies that the task needs more attention from the research community.

D. Cantone et al. (Italy) propose a corpus-based statistical measure of closeness between two sets of words. The measure is based on co-occurrence statistics of the words from the two sets, calculated over a large enough text corpus. The problem is important in a great number of text processing and computational linguistics-related tasks where semantic text similarity is used, ranging from information retrieval to plagiarism detection. The authors present computationally efficient algorithms for computing the proposed closeness measure.

M. Mouriño-García *et al*. (Spain) introduce the bag-of-concepts representation scheme for text processing and show its advantages of the traditional bag-of-words representation scheme. In particular, the use of concepts instead of words alleviates the problems related to synonymy and polysemy of words. Their experiments confirm this intuition; however, the results heavily depend on the quality of concept extraction method used to build the bag-of-concepts representation of the documents.

This issue of IJCLA will be useful for researchers, students, software engineers, and general public interested in natural language processing and its applications.

ALEXANDER GELBUKH
EDITOR IN CHIEF

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 11–24
Received 23/02/2015, Accepted 17/05/2015, Final 15/06/2015.
ISSN 0976-0962, http://ijcla.bahripublications.com

Identifying Linguistic Correlates of Social Power

RACHEL COTTERILL [1]
KATE MUIR [2]
ADAM JOINSON [2]
NIGEL DEWDNEY [1]

[1] *University of Sheffield, Sheffield, UK*
[2] *University of the West of England, Bristol, UK*

ABSTRACT

Previous work on social power modelling from linguistic cues has been limited by the range of available data. We introduce a new corpus of dialogues, elicited in a controlled experimental setting where participant roles were manipulated to generate a perceived difference in social power. Initial results demonstrate successful differentiation of upwards, downwards, and level communications, using a classifier built on a small set of stylistic features.

1. INTRODUCTION

One of the fastest growing areas of computational sociolinguistics in recent years has been the task of inferring various personal attributes from linguistic data. This is a popular mechanism for making sense of the social web and its ever-increasing quantities of data. Studies have spanned a range of topics, including classification by age, gender, native language, social group membership, and mental state.

The task of categorising relationships is a particularly interesting instance of the general problem. Unlike a demographic attribute such as age or native language, which is relatively stable for an individual across all communicative

contexts, we expect to see the same individual participating in a range of different social roles and relationships: the speaker's production is directly influenced by the specific audience.

The category of interpersonal relationships that has recieved the majority of scholarly attention to date is in the arena of hierarchy and social power, in part because this is a comparatively well-defined relation which is typically codified within an organisational structure. For example, managers are generally assumed to sit above their staff in the social hierarchy, professors are senior to students, and forum moderators have a position of power over ordinary contributors.

The majority of previous studies on categorising relationships and identifying power have relied on existing datasets such as Enron emails [1, 2], discussions between Wikipedia editors [3], and courtroom transcripts [4]. These studies have highlighted the shortage of publicly available datasets with high-quality ground truth. For example, Enron studies have made use of sparse hierarchies, reconstructed from publicly available information on organisational roles; these cover only a small subset of the individuals represented in the data, and do not form a well-connected graph [5, 6]. In the rare cases where experimental data has been gathered (e.g. [7]), these datasets have not been published, rendering them of limited use to the wider community.

This paper introduces a new, public dataset of transcribed speech, gathered in an experimentally controlled setting. We use this data to study the stylometric expression of social hierarchy.

2. PREVIOUS WORK

The effect of hierarchy and power on linguistic choices has always been of interest to linguists and sociologists. Brown & Levinson's [8] politeness theory identified relative power (the asymmetric relation) as one major factor of politeness in language, alongside social distance (the symmetric relation) and degree of imposition.

In more recent studies, computational approaches have examined qualitative approaches to large data sets. Peterson et al.

[5] investigate the applicability of Brown & Levinson's theory to email data, looking for correlations between informal features in text, and the level of politeness predicted by the theory. The features which they use to identify informal text include informal word lists, punctuation features (such as use of exclamation marks, or missing sentence-final punctuation), and case features (such as lowercase sentences). They report that informality features in the Enron email corpus are distributed largely as predicted by politeness theory.

Danescu-Niculescu-Mizil, et al. [3] study politeness within two online datasets: discussions between Wikipedia editors, and on Stack Exchange. They use Mechanical Turk to annotate turns with level of politeness, and demonstrate a distribution of politeness features in line with Brown & Levinson's predictions. They show that politeness is a precursor to promotion, at least in a community-approval model such as becoming an admin for Wikipedia: users who employ more politeness strategies are more likely to succeed in their social goals, and subsequently become less polite following promotion.

In another study of the Enron corpus, Bramsen et al [1] build an n-gram model and report a classification accuracy of 78.1% on the upspeak-downspeak task, and 44.4% accuracy on the three-way task of distinguishing upwards, downwards, and level communications. Cotterill [2] builds on Bramsen et al.'s work to model social power using only stylistic features, achieving comparable results with a smaller feature set.

Gilbert [9] examines the manifestation of power in the Enron corpus from a phrase-based perspective, using penalized logistic regression to identify those phrases which are particularly correlated with high or low power (as defined by job roles within the company). Using an SVM classifier to measure the predictivity of the resulting features, he reports an accuracy of 70.7% under three-fold cross validation.

Kacewicz et al. [7] undertake a series of five experiments with social power manipulation under different conditions, and report generalised findings relating to the differing use of pronouns. Lower-status individuals were observed to use more first person singular forms, while first person plural was used

more commonly by higher-status individuals. Second person forms were also used more by higher-status speakers, although the difference was less marked in this case.

3. DATA ELICITATION

We recorded and transcribed a collection of dyadic interactions as part of an applied psychology experiment into power-differential behaviour in a simulated business environment.

Volunteers were recruited from the student body at [anonymised] and given a task to complete, which they were advised concerned "creativity in business." A total of 41 participants took part in the study. The experimental group was composed of twelve participants assigned to the "judge" role and twelve "workers" (after [10]). The remaining 17 participants were assigned to the control condition.

In the experimental group the participants were randomly divided into judges and workers. The workers were given brief outlines of product ideas: these were drawn from Kickstarter campaigns, and featured an image and a short product description text. The workers pitched each idea to a judge, in a one-to-one conversation, and following a brief period of discussion the judges then chose whether or not to 'invest' in the concept. Both sets of participants were given to understand that the judges' ratings would affect the level of payment received by the workers for their participation, whereas the workers were given no such mechanism to provide feedback on the judges, thereby generating a scenario with a clear power differential between the two groups. (To satisfy the ethics board, eventual payment was in fact at a fixed rate for all participants.)

Members of the control group were similarly divided into two groups and provided with idea sheets, but instead of a worker/judge dynamic they were asked to discuss the inventions between themselves with an eye to potential collaborations. Neither party was given a higher status in the interaction, and they were informed that their participation payment would be a fixed amount, regardless of interaction success.

In both conditions, participants rotated through multiple conversation partners using a "speed dating" model to generate a number of independent one-to-one interactions lasting five minutes each. These exchanges were recorded, and after the end of the experiment the recordings were professionally transcribed. With a couple of exceptions due to corrupted files, one interaction was recorded between each judge/worker pair in the experimental condition (142 conversations) and between each pair in the control condition (72 conversations). The recorded conversations sum to 13,266 turns, giving a mean of 61.99 turns per dyad. The distribution of turns varied between the hierarchical ($\mu = 59.92$, $\sigma = 29.23$) and non-hierarchical ($\mu = 66.07$, $\sigma = 20.30$) condition, but this does not represent a statistically significant variation.

From a sociolinguistic perspective, the major disadvantage of this dataset is that it does not contain example utterances from the same individual participating under more than one role. A given student took on the role of judge, or worker, or part of the control group, and maintained this role for the duration of the experiment. It is therefore not possible to measure how an individual's linguistic choices shift in response to the changing of their relative power within a scenario.

A range of supplementary data was collected from each participant, including demographic information and personality profiling questionnaires. Most of the participants (82.9%) were undergraduate students from the University of [anonymised]. The remainder was made up of postgraduate students and non-students. Participants' ages ranged from 18 to 25. Female subjects made up 70.7% of the population, and 75.6% listed their ethnic origin as British.

As the data was elicited under controlled circumstances, we have reliable information concerning which participants were assigned to which social roles. The participants did not know one another in advance, so unlike in genuine organisational contexts, it is not necessary to account for the possibility of existing social relationships crossing these hierarchical boundaries in unexpected ways. As the roles were assigned at random, we also avoid the possibility of interference from underlying personality

traits or other demographic factors, which might lead to someone achieving a leadership role while also being expressed via their language choices.

With an experimental setup, there is always a risk that the participants' behaviour may be affected by the artificial nature of the setting. However, as we will demonstrate, the data still exhibits significant stylistic differences between speakers in different roles. After the experiment, a manipulation check was conducted by asking participants to score the level of power they felt they had during the interactions: results indicated that judges felt the most powerful ($\mu = 3.7$, $\sigma = 1.1$), while workers reported lower scores ($\mu = 2.8$, $\sigma = 1.1$), which is significantly different at 95%. Interestingly, both control groups rated their perceived power as less than either of the experimental groups ($\mu = 1.8$, $\sigma = 1.1$ and $\mu = 2.0$, $\sigma = 1$), which may be a consequence of participating in a scenario where their actions were not expected to change any of the outputs.

4. CLASSIFYING SOCIAL POWER

4.1. *Feature selection*

Following earlier work on social power modelling, we select a set of stylistic features to model our data. For email data, stylistic features have been shown to be broadly as effective as n-gram features, while resulting in a model of significantly lower dimensionality [2]. We apply an equivalent feature set, while noting that speech data lacks a number of the features that would be indicative of informality in text, such as varying capitalization or innovative punctuation.

One particular advantage of stylometrics is that selection of stylistic features tends to be subliminal: for example, in spontaneous production, an individual cannot control his use of function words such as pronouns or determiners.

A full list of features is included in Table 1. The majority of these are self-explanatory, but some would benefit from further elucidation.

Table 1. *List of stylometric features*

Characters per word	Interjections
Words per sentence	Expletives
Sentences per utterance	Contractions
Commas	Polite expressions
Periods	Hedging expressions
Semicolons	Deictic expressions
Colons	Modal verbs
Question marks	Verbs
Exclamation marks	Nouns
Hyphens	Pronouns
Parentheses	Determiners
Uppercase letters	Adjectives
Tag questions	Adverbs
Heylighen-Dewaele F-score	Prepositions
Out-of-vocabulary words	Conjunctions
Numbers	

Because the data has been professionally transcribed, there is less chance of typographical errors, contrasted with text that has been spontaneously produced – and if such errors do exist, they are due to the transcriber rather than the participant. Nevertheless, a measure of out of vocabulary words (measured with respect to an English dictionary) may prove a valuable feature as this encompasses a number of phenomena including codeswitching, informal slang, and highly technical jargon.

We retain the distribution of punctuation as a feature set, on the assumption that the transcriber's selection of punctuation will reflect speech-related features such as timing and pitch. Similarly, the concept of a 'sentence' in speech is controversial, but we nevertheless retain it as a feature for comparison with earlier work. The distribution of uppercase letters is also employed as a useful proxy for proper nouns (encompassing some such as product names which may not be captured by an entity tagger).

Parts of speech are tagged using the OpenNLP toolkit. Heylighen and Dewaele's F-score [11] is a linear combination of parts of speech, following a formal definition of contextuality; this is included as a separate feature.

4.2. *Individual message results*

A random forest classifier (using WEKA) was trained over the stylistic features from Table 1, and performance was assessed using five-fold cross validation. The logical baselines for this task are the random baseline, at 33.3%, and the most common class (level) 35.86%.

Message-level accuracy was 41.98%, using all features. Broken down further, this represents 36.59% accuracy for messages going up the hierarchy, 40.79% for downwards messages, and 47.87% accuracy for messages that formed part of peer-level exchanges.

From the resulting confusion matrix it is evident that level communications are the most successfully classified, but at the cost of classifying a number of upwards and downwards messages into the 'level' category.

Table 2. *Confusion matrix: message level results. Columns are predicted values, rows are truth*

	Upwards	Downwards	Level
Upwards	**1556 (11.7%)**	1144 (8.6%)	1553 (11.7%)
Downwards	1081 (8.1%)	**1736 (13.1%)**	1439 (10.8%)
Level	1217 (9.2%)	1263 (9.5%)	**2277 (17.2%)**

It is also interesting to consider individual variation. Classification accuracy at the individual level (calculated across all messages sent by that individual) ranges between 16.6% and 69.3%, following an approximately normal distribution (μ = 42.7, σ = 11.0). From this we can see that some individuals use language in a way that is 'more typical' of their role, while others are more divergent in their linguistic behaviour.

Our results at this stage are above baseline performance, although a couple of percentage points below the results reported for email in [1] and [2]. This is clearly unlikely to represent sufficient performance for any real-world applications, so we will proceed to examine ways in which accuracy can be enhanced.

4.3. *Simple plurality voting*

So far, we have considered categorisation at the message level, with results that are promising but not groundbreaking. However,

it is unlikely that any individual message perfectly captures the entire essence of a pair's relationship, and as such, we might expect to get better results by combining predictions from multiple messages.

There are two distinct methods for approaching such a task: a classifier can be trained on the aggregate features of the whole message set, or the results of single-message classification can be combined in an additional step. Since we have already obtained above-chance performance at the single-message level, we adopt the second approach.

The most basic method of combining scores is to use a 'voting' method. For example, given a set of twenty messages between A and B, we might have the following output from our individual message classification:

A to B	upwards: 7	downwards: 2	level: 1
B to A	upwards: 4	downwards: 6	level: 0

Based on these numbers, we would have one vote for an equal relationship, and 19 for a hierarchy. Looking further into the hierarchical evidence, we find 7 + 6 = 13 votes for A being subordinate to B, and 2 + 4 = 6 votes for B being subordinate to A. In this case, if A is indeed B's subordinate, we have the potential to turn 65% message-level accuracy into a single correct prediction at the relationship level. Of course, the inverse of this is that when we get it wrong, we will be degrading our overall performance.

For our initial experiments with aggregation, we simply took as our answer whichever case had the highest number of votes in total. We will refer to this technique as 'simple plurality voting', by analogy with electoral systems such as first-past-the-post.

Considering aggregation at the level of the individual speaker, applying simple plurality voting to the classifier output gives us two predictions for each dyad, one based on each speaker's output. We assess the accuracy of these predictions independently, and observe that our overall mean accuracy increases to 57.9% at the speaker level — but variance also increases, from 11.0 to 27.0, and our distribution is no longer

normal. Figure 1 demonstrates this shift in distribution. Note that we have 41 speakers producing 13,266 turns: we display results as percentages for ease of comparison, but in absolute terms, all numbers are much smaller in the aggregated case.

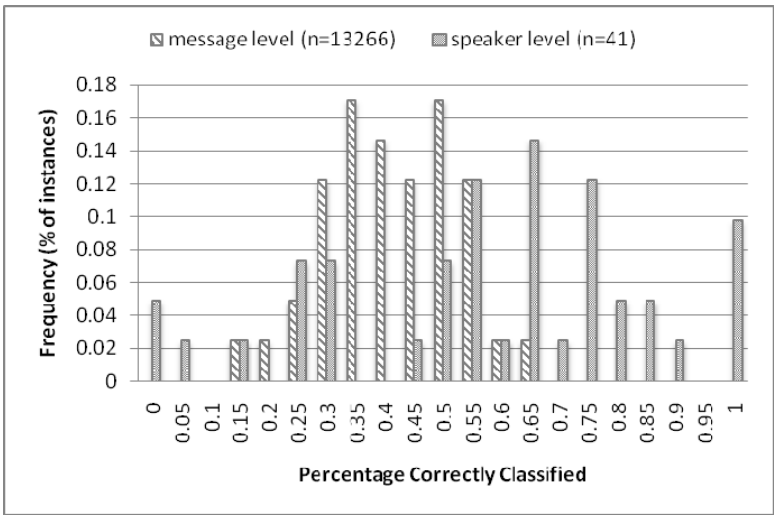


Figure 1. *Chart demonstrating the distribution of correctly-classified instances for individual messages, and for speaker-level aggregation.*

Speaker-level aggregation is simple and informative, but still leaves us with two predictions for each dyadic relationship, which may be in conflict. We extend the simple plurality voting method to include votes in both directions, as a single system set up to generate one prediction per pair. Due to the collection methodology of the experimental dataset we are guaranteed, for each dyad, an approximately equal number of utterances in each direction, so in this instance there is no need to concern ourselves with imbalances in the data.

Using simple plurality voting on a pairwise basis we achieve 69.1% accuracy in the task of three-way prediction across pairs, with seven pairs unassigned (cases where there was no single 'most common' class).

The resulting error analysis shows that the system is more likely to mis-categorise relationships as level when they are actually hierarchical; by comparison, incorrectly inverting the hierarchy is relatively rare.

Table 3. *Error analysis: Pairwise aggregation (correct lines in bold). Percentages exclude the seven uncategorised instances*

82	**39.61%**	**Hierarchical, correctly labelled**
45	21.74%	Hierarchical, incorrectly labelled as level
10	4.83%	Hierarchical, labelled with incorrect polarity
61	**29.47%**	**Level, correctly labelled**
9	4.35%	Level, incorrectly labelled as hierarchical

4.4. *The effect of thresholds*

Intuition suggests that it should be possible to obtain a higher degree of accuracy by setting a minimum confidence threshold, and accepting classifications only above this threshold.

One simple method of applying a threshold to a voting system is to set a minimum percentage of messages which must fall into the 'most popular' classification before it can be accepted. For a three-class problem such as this, the default (and lowest possible) threshold for a simple plurality vote is 0.33, as it isn't possible for all three classes to obtain less than a third of the available votes.

We investigated setting higher thresholds, from 0.4 up to 0.6. Increasing the threshold gives an almost linear improvement in raw accuracy (over the classified instances), but at the cost of rejecting ever higher number of instances without classification. The improvement in precision comes with a fairly steep drop in recall once the threshold is above 0.4. As always, the appropriate compromise between precision and recall will vary depending on the application.

Table 4. *Effect of thresholding on precision and recall*

	0.333	**0.4**	**0.5**	**0.6**
Unclassified pairs	7	25	109	177
Accuracy (%)	69.08	71.96	75.24	78.37

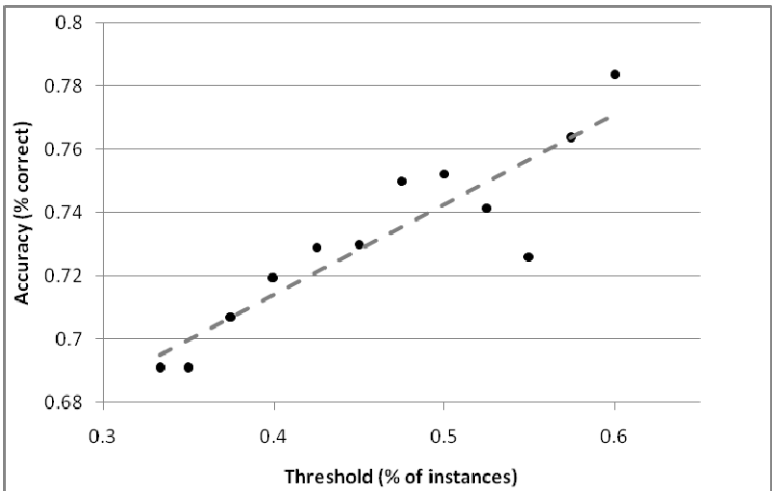


Figure 2. *Trend of accuracy as threshold is raised*

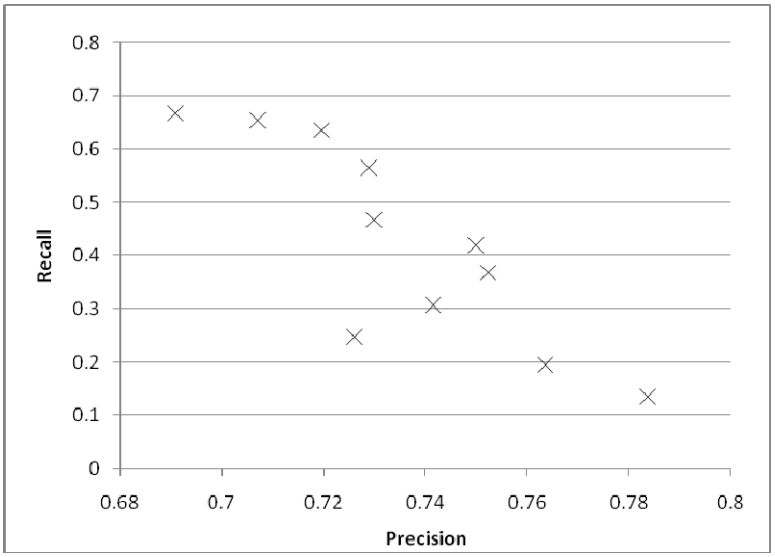


Figure 3. *Precision-recall plot for varying confidence thresholds*

5. Conclusions and Future Work

Using a small set of stylistic features, we have achieved above-chance performance at the individual message level for classifying spoken dialogues. Additionally, we have demonstrated a significant improvement in performance as a direct result of aggregating data at the relationship level. We have shown that introducing a threshold can improve precision, but only at the cost of a significant drop in recall, which is unlikely to be a worthwhile trade-off in real world applications.

In future work we intend to address a number of limitations of our experimental set-up. We plan to replicate our data collection step using a computer-mediated setting, to allow for direct comparison of spoken and textual conversations. Additionally, we hope to design a suitable scenario which would allow for the possibility of participants participating under more than one role.

References

1. Bramsen, P., Escobar-Molano, M., Patel, A. & Alonso, R. 2011. Extracting social power relationships from natural language. In proceedings of the *Annual Meeting on Association for Computational Linguistics* (pp. 773-782).
2. Cotterill, R. 2013. Using stylistic features for social power modeling (El uso de características estilísticas para modelado del poder social). *Computación y Sistemas*, 17/2, 219-227.
3. Danescu-Niculescu-Mizil, C., Sudhof, M., Leskovec, J. & Potts, C. 2013. A computational approach to politeness with application to social factors. In *Proceedings of ACL*.
4. Danescu-Niculescu-Mizil, C., Lee, L., Pang, B. & Kleinberg, J. 2012. Echoes of power: Language effects and power differences in social interaction. In proceedings of the *21st International Conference on World Wide Web* (pp. 699-708), New York, NY, USA.
5. Peterson, K., Hohensee, M. & Xia, F. 2011. Email formality in the workplace: A case study on the Enron corpus. In proceedings of the *Workshop on Languages in Social Media* (pp. 86-95).

6. Agarwal, A., Omuya, A., Harnly, A. & Rambow, O. 2012. A comprehensive gold standard for the Enron organizational hierarchy. In proceedings of the *50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2* (pp. 161-165), Stroudsburg, PA, USA.
7. Kacewicz, E., Pennebaker, J. W., Davis, M., Jeon, M. & Graesser, A. C. 2013. Pronoun use reflects standings in social hierarchies. *Journal of Language and Social Psychology*, Sep.
8. Brown, P. & Levinson, S. C. 1987. *Politeness: Some Universals in Language Usage*. Cambridge: Cambridge University Press.
9. Gilbert, E. 2012. Phrases that signal workplace hierarchy. In proceedings of the *ACM 2012 Conference on Computer Supported Cooperative Work* (pp. 1037-1046), New York, NY, USA.
10. Guinote, A., Judd, C. M. & Brauer, M. 2002. Effects of power on perceived and objective group variability: Evidence that more powerful groups are more variable. *Journal of Personality and Social Psychology*, 82/5, 708-721.
11. Heylighen, F. & Dewaele, J. M. 2002. Variation in the contextuality of language: An empirical measure. *Foundations of Science*, 7, 293-340.

RACHEL COTTERILL
UNIVERSITY OF SHEFFIELD, SHEFFIELD, UK.
E-MAIL: < R.COTTERILL@SHEFFIELD.AC.UK>

KATE MUIR
UNIVERSITY OF THE WEST OF ENGLAND, BRISTOL, UK.
E-MAIL: < KATE.MUIR@UWE.AC.UK>

ADAM JOINSON
UNIVERSITY OF THE WEST OF ENGLAND, BRISTOL, UK.
E-MAIL: < ACP08NJD@SHEFFIELD.AC.UK>

NIGEL DEWDNEY
UNIVERSITY OF SHEFFIELD, SHEFFIELD, UK.
E-MAIL: < ADAM.JOINSON@UWE.AC.UK>

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 25–44
Received 28/02/2015, Accepted 18/05/2015, Final 12/06/2015.
ISSN 0976-0962, http://ijcla.bahripublications.com

On the Influence of Text Complexity on Discourse-Level Choices

ELNAZ DAVOODI
LEILA KOSSEIM
Concordia University, Montreal, Canada

ABSTRACT

Text complexity can be reduced by making different choices at the lexical and grammatical levels. However, discourse-level choices may also affect a text's complexity. In a coherent text, explicit discourse relations (e.g. CAUSE, CONDITION*) are expressed using discourse markers (e.g.* since, because*, etc.) that may be preferred for texts at different readability levels. In this paper, we investigate the differences in discourse properties of texts across readability levels. In particular, we investigate the effect of readability level on (1) the usage of discourse relations, (2) the usage of discourse markers and (3) the distribution of discourse markers signaling explicit discourse relations. Our analysis of the Simple English Wikipedia corpus shows that complex and simple texts seem to have the same distribution of discourse relations; however, these relations are expressed using different discourse markers depending on the readability level of the text.*

1. INTRODUCTION

In well-written texts, utterances are connected to each other using Discourse Relations (DRs) which allow the reader to understand the communicative intention of the writer. DRs (e.g. CAUSE, CONDITION) can be expressed either implicitly or explicitly. *Implicit* relations are not signalled using lexical cues such as *but*, *since*, *because*, etc. and must be inferred by the readers. On the other hand, *explicit* relations are signalled using specific terms

called Discourse Markers (DM). According to [21], DMs constitute strong clues to detect explicit relations, hence discourse parsers have used them as valuable features in order to identify DRs automatically (e.g. [13, 26, 10, 16]).

A text's discourse-level properties have been shown to be correlated to various dimensions such as their genre, their level of formality, their level of readability, etc. For example, Webber [27] and Bachand et al. [1] showed that the textual genre influences the choice of DRs. In order to produce texts at various readability levels, several techniques have been proposed to simplify texts at the lexical level (e.g. [7, 30]), the syntactic level (e.g. [3, 24]) and the discourse level (e.g. [25]). In particular, Williams [28] used "simpler" DMs to generate more readable texts for people with a lower level of literacy. In the process of text simplification, the writer's goal is to reformulate a text to make it easier to read and understand; however, its informational content should be preserved. Based on this assumption, we suspected that the simplification process should not change the semantic or logical relations between textual units; however, because DRs can be used for rhetorical purposes [17], the distribution of DRs may be different across text complexity.

One can view texts at different readability levels as translations of their "regular" counterpart. Using this perspective, we can argue that during the translation, translators may choose to use DRs and DMs differently in the translated text by adding or removing them or making implicit relations explicit or vice versa; all the while, preserving the meaning of the original text. For example, in the context of machine translation, Meyer and Webber [18] have shown that fewer DMs were used in the German or French translations of the Newstest 2012 parallel corpus[1] compared to its English counterpart.

In this article, we investigate the inuence of the readability level on the usage of explicit DRs and DMs. We have used the Simple English Wikipedia corpus [4] which has been used widely in text simplification and the related task of text

[1] http://www.statmt.org/wmt12/

compression (e.g. [29, 30]). The usage of explicit DRs and DMs as well as the distribution of DMs used as cues of such relations are analyzed. We used the log-likelihood ratio to rank the DRs and DMs with texts at various levels of readability.

This paper is organized as follows: Section 2 describes the corpus preparation to extract DRs and DMs. Section 3 presents the results of each experiment. Related work and discussions are presented in Section 4 and finally Section 5 presents our conclusions and future work.

2. CORPUS PREPARATION

To investigate the infuence of the readability level on the usage of DRs and DMs, these were extracted automatically from parallel corpora accross different readability levels.

2.1. *The simple English Wikipedia corpus*

Because they are manually annotated with DRs, the RST-DT corpus [2] and the Penn Discourse TreeBank (PDTB) [22] constitute two of the most widely used corpora for discourse analysis. However, these corpora could not be used in our work because we needed a parallel corpus across different readability levels. Instead, we used the Simple English Wikipedia corpus [4] which is a parallel corpus containing regular and simplified versions of Wikipedia articles. The simplified versions of the Wikipedia articles are meant to be more accessible to beginners learning English, such as students, children, adults with learning difficulties and people who are trying to learn English. These articles are typically shorter than their regular counterparts, and use simpler words and syntactic structures. The simplified articles were created by using their regular counterparts as a basis and following a set of simplification guidelines.[2] In particular, word choices are limited to Basic English[3], a 850-word auxiliary

[2] https://simple.wikipedia.org/wiki/Wikipedia: How to write Simple English page

[3] https://simple.wikipedia.org/wiki/Wikipedia:Basic English ordered wordlist

international language, and the VOA Special English Word Book,[4] a list of 1580 words. The guidelines are not only limited to lexical choices, but also suggest the use of simpler syntactic structures; such as avoiding compound sentences containing embedded conjunctive clauses.

The Simple English Wikipedia corpus was first created from Simple Wikipedia articles[5] in 2010. The first version of this corpus contains 137K aligned sentences pairs created from Wikipeda pages downloaded in May 2010. The latest version, released in 2011, contains two parts: a sentence-aligned part containing 167K aligned sentence pairs and 60K aligned articles. In our work, we used the aligned sentences of the latest version of this corpus.

2.2. *Labeling the corpus*

Because the Simple English Wikipedia corpus is not discourse-annotated, to label DRs and identify DMs signalling explicit DRs, we have automatically parsed the parallel sentences using the End-To-End PDTB-based discourse parser [16].

Several other publicly available discourse parsers could have been used (eg. [13, 10, 9]). We chose the End-to-End parser because we needed local discourse-level information that include the type of discourse relations (i.e. *implicit* or *explicit*), the name of the discourelation and the discourse marker when applicable. When the work was performed, the End-to-End parser was the best performing parser providing all these features. Although the parser can identify both explicit and implicit DRs, we only considered explicit DRs as the accuracy of the parser in detecting explicit relations is about 81.19% whereas for implicit relations the accuracy drops significantly. In addition, because we are interested in the usage of discourse markers which signal explicit DRs, implicit relations were not considered.

---

[4] https://simple.wikipedia.org/wiki/Wikipedia:VOA Special English Word Book

[5] www.simple.wikipedia.org

The End-to-End parser [16] uses the PDTB inventory of relations [22] organized into 3 levels of granularity. Level 1 includes four relations: TEMPORAL, CONTINGENCY, COMPARISON and EXPANSION. In our experiment, we used the 2nd level that defines 16 relations, but only 12 relations were present in the corpus. In addition, the End-to-End parser uses an inventory of 100 DMs, but only 72 were actually present in the Simple English Wikipedia corpus.

Table 1 provides statistics about the annotation of the regular and simple versions of the Simple English Wikipedia corpus with the End-to-End parser. As shown in Table 1, the regular version of the sentence-aligned part of the corpus contains 167K sentences; however in the simple version, the number of sentences increases to 189K sentences. In the simple version, sentences tend to be shorter (18.45 words versus 23.36) and fewer DMs are used. In addition, the ratio of DM per token is tend to be lower in the simple version compared to the regular version (0.093 vs 0.098).

Table 1. *Statistics of the Simple English Wikipedia corpus*

	Regular version	**Simple version**
# of sentences	167,690	189,572
# of DMs	52,648	48,412
token/sentence ratio	23.36	18045
DM/token ratio	0.098	0.093
DM/sentence ratio	0.31	0.25

3. ANALYSIS

Once the Simple English Wikipedia was tagged with DMs and DRs, we analysed: (1) the usage of DRs, (2) the usage of DMs and (3) the distribution of DMs over DRs across readability levels.

3.1. *Effect of text complexity on the usage of DRs*

Once the parallel corpus was parsed with End-to-End parser [16], we extracted the explicit DRs in both the regular and the simple versions. In order to eliminate the effect of corpus size, we

considered the relative frequencies of DRs, then we performed frequency profiling using the log-likelihood ratio [23]. This measure allows us to compare the frequency of DRs across the regular and the simple versions and sort them according to the importance of their relative frequencies. The log-likelihood ratios themselves only provide a measure of which DRs are statistically more informative. The results are shown in Table 2 in decreasing order of log-likelihood ratio. The relations at the top of the table are therefore more indicative of the regular version, as compared to the simple versions of the corpus.

According to Table 2, the most differences stem from the relations of CONTRAST, CAUSE and CONCESSION; however in both the regular and the simple versions, the three most frequent DRs are CONJUNCTION, CONTRAST and ASYNCHRONOUS.

In order to verify if these changes are statistically significant, we first performed a normality test using the IBM SPSS software[6] to investigate the characteristics of our data set. According to this test, the relative frequency of DRs in the regular and simple versions is not normally distributed. Consequently, we have used the Wilcoxon test of statistical significance to see if the difference across the two corpora are statistically significant. The Wilcoxon test is a non-parametric statistical hypothesis test which is an alternative to the Student's t-test when the population is not normally distributed. According to this test, the differences in the relative frequencies of DRs are not statistically significant. As a result, we can conclude that the usage of DRs seems to be preserved across different readability levels of this parallel corpus.

[6] http://www-01.ibm.com/software/analytics/spss/

Table 2. *Relative frequency of DRs across regular and simple versions of the Simple English Wikipedia corpus sorted by log-likelihood ratio*

Discourse Relation	Regular Version	Simple Version	LL Ratio
CONTRAST	18.10%	16.29%	20.76
CAUSE	7.62%	8.64%	13.82
CONCESSION	2.88%	2.33%	12.49
RESTATEMENT	0.31%	0.20%	4.85
CONDITION	4.06%	4.46%	4.01
ASYCHRONOUS	14.76%	15.31%	2.22
SYNCHRONY	12.51%	12.75%	0.48
EXCEPTION	0.04%	0.05%	0.22
LIST	0.01%	0.02%	0.17
CONJUNCTION	36.52%	36.72%	0.12
ALTERNATIVE	1.75%	1.78%	0.06
INSTANTIATION	1.38%	1.39%	0.00

3.2. *Effect of text complexity on the usage of DMs*

Given that the usage of DRs seems to be preserved, we next turned to how they are signalled across readability levels. DMs can signal more than one DRs. For example, *although* can signal both a CONCESSION and a CONTRAST relation. In this experiment, we were interested in investigating the distribution of DMs over DRs.

Once all the DMs and DRs were extracted using the End-to-End parser [16], we constructed DM/DR pairs in order to disambiguate DMs that can signal more than one DR. As a result, we created a set of 119 unique DM/DR pairs. Then, we again used log-likelihood ratios to sort the pairs. Hence, a DM/DR pair with a higher log-likelihood ratio is more indicative of the regular version, as compared to the simple version of the corpus. Table 3 shows the 10 most discriminating pairs across the regular and simple versions.

Using all DM/DR pairs extracted automatically, we have again performed a statistical significance test in order to determine if the difference in the relative frequency of DM/DR pairs across corpora is statistically significant. Similarly to the first analysis (see Section 3.1), we first performed a normality test using the IBM SPSS software. The results revealed that

DM/DR pairs are not normally distributed across corpora. The relative frequency of some pairs such as *because*/CAUSE, *so*/CAUSE and *but*/CONTRAST is higher in the simple version, while it is lower for other pairs such as *thus*/CAUSE, *although*/CONTRAST and *while*/CONTRAST. The Wilcoxon statistical significance test showed that the relative frequency of DM/DR pair across different readability levels is statistically different. More precisely, the Wilcoxon test revealed that in the simple version of the Simple English Wikipedia, DMs are used less frequently than in its regular counterpart. This is an interesting finding as it seems to indicate that to make a text more accessible, the use of discourse markers should be reduced; hence not indicating discourse relations explicitly.

Table 3. *Relative frequency of DM/DR pairs across regular and simple versions of the Simple English Wikipedia corpus sorted by log-likelihood ratio*

Discourse Relation	Regular Version	Simple Version	LL Ratio
because/CAUSE	0.0280%	0.0470%	76.99
thus/CAUSE	0.0166%	0.0094%	38.79
although/CONTRAST	0.0211%	0.0134%	33.96
so/CAUSE	0.0206%	0.0311%	32.89
while/CONTRAST	0.0433%	0.0331%	28.84
when/SYNCHRONY	0.0766%	0.0955%	27.92
also/CONJUNCTION	0.2088%	0.2398%	24.35
as/SYNCHRONY	0.0564%	0.0474%	18.22
although/CONCESSION	0.0216%	0.0160%	17.52
but/CONTRAST	0.0760%	0.0902%	15.12

3.3. *Effect of text complexity on the distribution of DMs over DRs*

Once we determined that there is a difference in how DMs are used to signal a DR across corpora, we tried to verify if the distribution of DMs to signal different DRs is different across readability levels. For example, as shown in Figure 1, the DM *while* can be used in the Simple English Wikipedia to signal two DRs: CONTRAST and SYNCHRONOUS. The following examples show sentences where the DM *while* signals a CONTRAST (sentence 1); whereas in sentence 2, it signals a SYNCHRONOUS relation.

1. While [any form of energy may be conserved], [electricity is the type most commonly referred to in connection with conservation.]/CONTRAST
2. [He began his career in primary education] while [an undergraduate teaching at the Children's Community School]/SYNCHRONOUS

In the regular version of the Simple English Wikipedia corpus, each DM conveys on average 1.68 relations. On the other hand, this number decreases to 1.61 in the simple version of the same corpus. As [14] noted, in the PDTB corpus, implicit and explicit DMs combined convey on average 3.05 relations. If we only consider explicit DRs, as in our work, this number decreased to about 2.6 in the PDTB. Because of this ambiguity of DMs, we wanted to investigate how specific DMs are used to signal different DRs across different readability levels. To do so, we identified the set of relations that each DM conveys, then, the distribution of all DMs across regular and simple versions has been computed. We have used entropy in order to calculate the information of each distribution; then, used cross entropy to measure the difference between the distributions [6, 11, 5]. Formula 1 is used to calculate the entropy of the distribution of each DM (noted as H(x)) across different readability levels. Each DM is considered as a random variable, the formula. The range of values that *x* can take, noted as r_i in Formula 1, are the possible DRs that the DM can signal. For example, using the DM *while* of Figure 1, the DM *x* is *while*, p(r_1) is the probability that the DM *while* is used to signal the CONTRAST relation which is 0.706 in the regular version as opposed to 0.676 in the simple version. Similarly, p(r_2) is the probability that the DM *while* signals a SYNCHRONOUS relations.

$$H(x) = H(p) = -\sum_i p(r_i) log(p(r_i)) \quad (1)$$

Discourse Relation	Probability in Regular version	Probability in Simple version
CONTRAST	0.706	0.676
SYNCHRONOUS	0.291	0.322

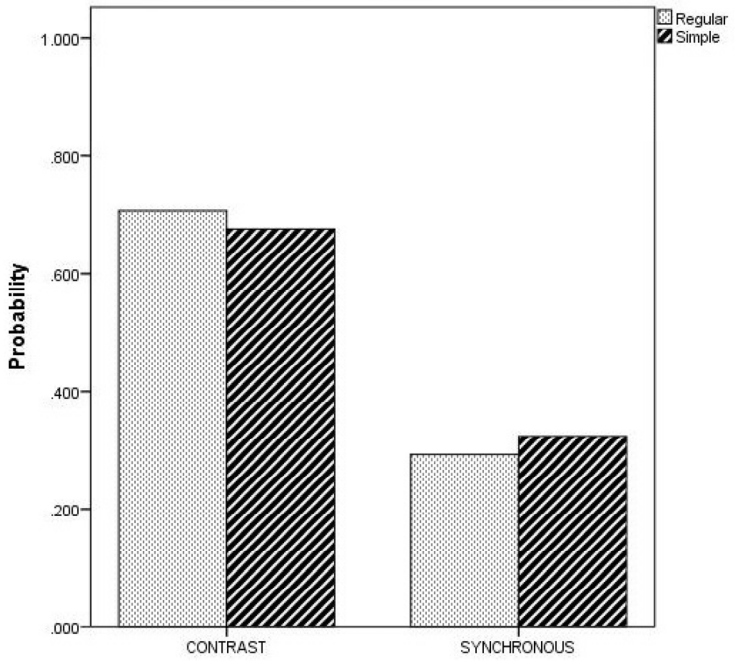


Figure 1. *Distribution of the DM while with respect to the DRs it signals across the Simple English Wikipedia corpus*

Once the entropy of each distribution has been computed, we have compared them in order to evaluate if there is a significant change in the distribution of DMs. To do so, we have used cross entropy. Formula 2 has been used for calculating the cross entropy for a specific DM called *x*. To compare two distributions using cross entropy, we assume that the first argument (*reg*) is the target probability distribution, and the other one (*simp*) is the estimated distribution that we are trying to compare against. The closer the cross entropy is to the entropy of the target distribution, the less the change in the distribution of the specific DM across readability levels. In our experiment, *reg* stands for

the regular version and $p((ri)_{reg})$ is the probability that the DM *x*, signalling the i^{th} relation in the regular version; while *simp* stands for the simple version and $p((x_i))_{\text{simp}}$ is the probability that the DM *x*, signals the i^{th} relation in the simple version.

$$H(reg, simp) = -\sum_i p((x_i)_{reg}) log(p((x_i))_{simp}) \quad (2)$$

The top 5 most differences in the distribution of DMs stems from the DMs in, *although, though, while* and *since*. Figure 2 shows the distribution of the DM in across the regular and the simple versions. In addition, the distribution of the DMs *although* and *though* both signalling CONCESSION and CONTRAST DRs are shown in Figures 3 and 4 respectively. As the figures show, both DMs are more frequently used to signal a CONCESSION in the simple version and a CONTRAST in the regular version. For example, the DM *although* is used 54.4% of the time to signal a CONCESSION in simple texts as opposed to 50.0% in regular texts. However, both *although* and *though* are more frequently used to signal a contrast in the regular version than in the simpler version. Finally, Figure 5 shows the distribution of the DM *since* to signal ASYNCHRONOUS and CAUSE DRs over the corpora. As Figure 5 shows, it is more probable that this DM is used to signal a CAUSE across both versions rather than ASYNCHRONOUS; however, to signal an asynchronous relation, it is more common to use since in the simple version than in the regular version.

It is interesting to note that although discourse relations seem to be preserved across readability levels (see Section 3.1), how discourse markers are used to signal these relations seems to vary across readability levels.

Discourse Relation	Probability in Regular version	Probability in Simple version
ASYNCHRONOUS	0.178	0.181
CAUSE	0.004	0.012
CONCESSION	0.002	0.003
CONJUNCTION	0.500	0.520
CONTRAST	0.093	0.062
INSTANTIATION	0.051	0.040
RESTATEMENT	0.170	0.178

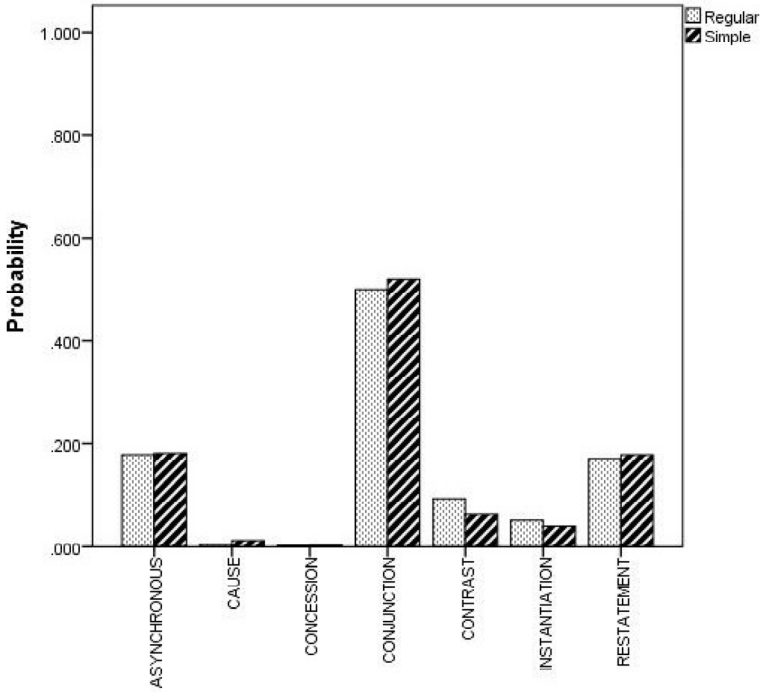


Figure 2. *Distribution of the DM in with respect to the DRs it signals across the Simple English Wikipedia corpus*

4. Related Work and Discussion

As [22] noted, DMs constitute valuable features to identify explicit DRs; however, they may be used in a non-discourse context. Several work have already addressed the identification, selection and placement of DMs in coherent texts (e.g. [12, 19, 8,

15, 20]). However, to our knowledge, no previous work has attempted to investigate the effect of readability level on the usage of DMs and DRs using large scale parallel corpora.

Discourse Relation	Probability in Regular version	Probability in Simple version
CONCESSION	0.506	0.544
CONTRAST	0.494	0.456

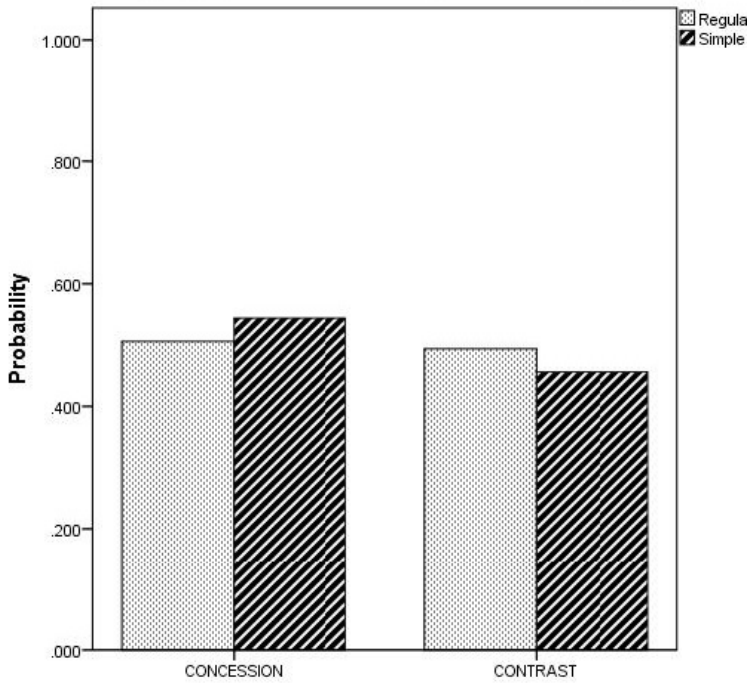


Figure 3. *Distribution of the DM although with respect to the DRs it signals across the Simple English Wikipedia corpus*

Discourse Relation	Probability in Regular version	Probability in Simple version
CONCESSION	0.394	0.483
CONTRAST	0.606	0.516

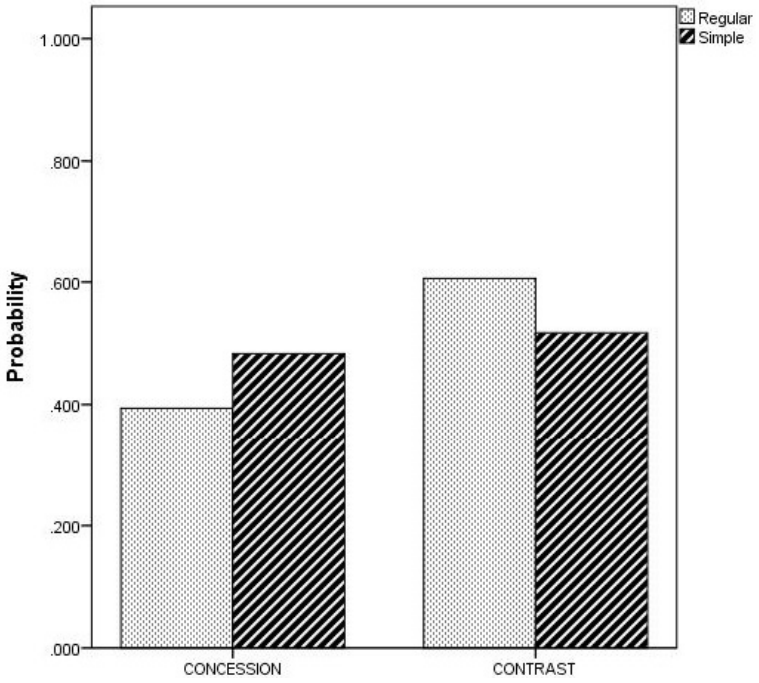


Figure 4. *Distribution of the DM though with respect to the DRs it signals across the Simple English Wikipedia corpus*

Discourse Relation	Probability in Regular version	Probability in Simple version
ASYNCHRONOUS	0.302	0.350
CAUSE	0.698	0.650

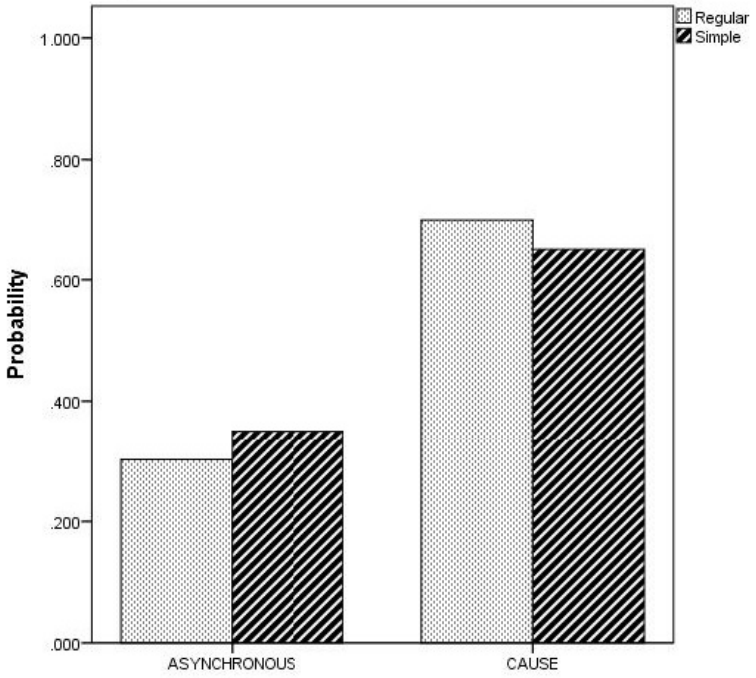


Figure 5. *Distribution of since with respect to the DRs it signals across the Simple English Wikipedia corpus*

Several attempts have been made to enhance the readability level of texts at different levels (i.e. lexical, syntactic or discourse levels) (e.g. [7, 30, 3, 24, 25]), or generating texts across different readability levels for various groups of audiences. For example, Williams' text generation system [28] generates texts at different levels of readability; however the simplification rules were based on a manual analysis of a small corpus. Three parallel texts (each with an average of 1000 to 2000 words) revealed some DMs like *so* and *but* are preferable to use in simpler texts than other DMs such as *therefore* or *hence*. She also reported that a more frequent usage of DMs result in more readable texts. This

last result seems to contradict our own (see Section 3.2) which are based on a much larger corpus.

Another related work is that of Siddharthan [25] who focused on textual simplification. Although the main focus of the work was on syntactic simplifications, Siddharthan also addressed the use specific DMs in order to increase the textual cohesion of the simplified texts. Once the original sentences were simplified syntactically, he selected specific DMs in order preserve the discourse relation between the resulting conjoined clauses. To do so, he used a set of 13 DMs and associated each DM to a single DR. The actual selection of the most appropriate DM was based on [28]'s recommendations. For example, every concession relation resulted in the use of the DM *but*. Although Siddharthan's main focus was not on discourse-level choices, a number of assumptions were made. In comparison, our work is based on a statistical analysis of a much larger corpus, uses a much larger set of DMs (the list of 100 DMs from the PDTB [22]) and does not assume a one-to-one correspondence between DMs and DRs.

5. Conclusion and Future Work

In this paper, we have performed an analysis of the usage of discourse relations (DRs) as well as the usage and distribution of discourse markers (DMs) across different readability levels. Our analysis of the Simple English Wikipedia corpus shows that discourse relations are preserved across different readability levels. However, the usage of discourse markers is different in the regular and their simpler counterparts. In particular, we observed that the relative frequency of DMs is higher in more complex texts. Additionally, our analysis revealed that the distribution of DMs to convey specific relations is different across different readability levels. These results seem to indicate that although the same logical and semantic information is conveyed in both simple and regular versions; how they are signalled is different.

In this article, we have analysed the changes in markers and relations at the document level, but did not look at individual

changes. As future work, it would be interesting to investigate discourse relations and discourse markers across specific sentence alignments in order to analyse changes in their individual usage. For example, under which conditions, a concession is changed to a condition at different readability levels.

REFERENCES

1. Bachand, F.H., Davoodi, E., Kosseim, L. 2014. An investigation on the inuence of genres and textual organisation on the use of discourse relations. In proceeding of the *15th International Conference of Computational Linguistics and Intelligent Text Processing (CICLing), LNCS-volume 8404*, (pp. 454-468). Springer.
2. Carlson, L., Okurowski, M.E., Marcu, D. 2002. RST discourse treebank. *Linguistic Data Consortium*, Catalog Number-LDC2002T07.
3. Chandrasekar, R. & Srinivas, B. 1997. Automatic induction of rules for text simplication. *Knowledge-Based Systems*, 10/3, 183-190.
4. Coster, W. & Kauchak, D. 2011. Simple English Wikipedia: A new text simplification task. In proceedings of the *49th Annual Meeting of the Association for Computational Linguistics*: *Human Language Technologies (ACL-HLT)* (pp. 665-669), Short papers-Volume 2. Portland, Oregon, June 2011)
5. Cover, T. M. & Thomas, J. A. *Elements of Information Theory*. John Wiley & Sons.
6. De Boer, P. T., Kroese, D. P., Mannor, S. & Rubinstein, R. Y. 2005. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134/1, 19-67.
7. Devlin, S. & Tait, J. 1998. The use of a psycholinguistic database in the simplification of text for aphasic readers. *Linguistic Databases* (pp. 161-173).
8. Di Eugenio, B., Moore, J. D. & Paolucci, M. 1997. Learning features that predict cue usage. In proceedings of *EACL* (pp. 80-87), Madrid, Spain.
9. Feng, V. W. & Hirst, G. 2012. Text-level discourse parsing with rich linguistic features. In proceedings of the *50th Annual Meeting of the Association for Computational Linguistics*: *Long Papers - Volume 1* (pp. 60-68), ACL-2012.

10. Hernault, H., Prendinger, H., A. duVerle, D. & Ishizuka, M. 2010. Hilda: A discourse parser using support vector machine classification. *Dialogue & Discourse*, 1/3.
11. Jaynes, E. T. 1957. Information theory and statistical mechanics. *Physical Review*, 106/4, 620.
12. Knott, A. 1996. A data-driven methodology for motivating a set of coherence relations. PhD dissertation, University of Edinburgh.
13. Laali, M., Davoodi, E. & Kosseim, L. 2015. The CLaC discourse parser at CoNLL-2015. In N. Xue, H. T. Ng, S. Pradhan, R. Prasad, C. Bryant, A. Rutherford, (Eds.), *Proceedings of the 19th Conference on Computational Natural Language Learning: Shared Task, CoNLL 2015* (pp. 56-60), Beijing, China, Jul 30-31.
14. Laali, M. & Kosseim, L. 2014. Inducing discourse connectives from parallel texts. In the *25th International Conference on Computational Linguistics (COLING)* (pp. 610-619), Dublin, Ireland.
15. Lin, Z., Kan, M. Y. & Ng, H. T. 2009. Recognizing implicit discourse relations in the Penn discourse treebank. In proceedings of the *2009 Conference on Empirical Methods in Natural Language Processing* (pp. 343-351), Volume 1, Singapore.
16. Lin, Z., Ng, H. T. & Kan, M. Y. 2014. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering*, 20/2, 151-184.
17. Mann, W. C. & Thompson, S. A. 1987. Rhetorical structure theory: A framework for the analysis of texts. Tech. rep., *IPRA Papers in Pragmatics*, 1.
18. Meyer, T. & Webber, B. 2013. Implicitation of discourse connectives in (machine) translation. In proceedings of the *1st DiscoMT Workshop at ACL 2013 (51st Annual Meeting of the Association for Computational Linguistics)* (pp. 19-26), Sofia, Bulgaria.
19. Moser, M. & Moore, J. D. 1995. Investigating cue selection and placement in tutorial discourse. In proceedings of the *33rd annual meeting on Association for Computational Linguistics* (pp. 130-135), Cambridge, Massachusetts, USA.
20. Patterson, G. & Kehler, A. 2013. Predicting the presence of discourse connectives. In EMNLP (pp. 914-923), Seattle, Washington, USA.
21. Pitler, E. & Nenkova, A. 2009. Using syntax to disambiguate explicit discourse connectives in text. In proceedings of the *ACL-IJCNLP 2009 Conference Short Papers* (pp. 13-16), Suntec, Singapore.

22. Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A. & Webber, B. L. 2008. The Penn discourse treebank 2.0. In proceedings of the *6th International Conference on Language Resources and Evaluation (LREC)*, Marrakesh, Morocco, Jun.
23. Rayson, P. & Garside, R. 2000. Comparing corpora using frequency profiling. In proceedings of the *Workshop on Comparing Corpora* (pp. 1-6), Hong Kong, Oct.
24. Scarton, C., De Oliveira, M., Candido Jr, A., Gasperin, C., Aluisio, S. M. 2010. Simplifica: A tool for authoring simplified texts in brazilian portuguese guided by readability assessments. In proceedings of *Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL): Demonstrations* (pp. 41-44), Los Angeles, CA, USA, Jun.
25. Siddharthan, A. 2006. Syntactic simplification and text cohesion. *Research on Language and Computation*, 4/1, 77-109.
26. Soricut, R. & Marcu, D. 2003. Sentence level discourse parsing using syntactic and lexical information. In proceedings of the *2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL)* (pp. 149-156), Vol. 1., Edmonton, Canada, Jun.
27. Webber, B. 2009. Genre distinctions for discourse in the Penn TreeBank. In proceedings of the *Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing (ACL-AFNLP)* (pp. 674-682), Vol. 2., Suntec, Singapore, Aug.
28. Williams, S., Reiter, E. & Osman, L. 2003. Experiments with discourse-level choices and readability. In proceedings of the *9th European Workshop on Natural Language Generation (ENLG-2003)* (pp. 127-134), Budapest, Hungary, Apr.
29. Yamangil, E. & Nelken, R. 2008. Mining wikipedia revision histories for improving sentence compression. In proceedings of the *46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies (ACL-HTL): Short papers* (pp. 137-140), Columbus, Ohio, Jun.
30. Yatskar, M., Pang, B., Danescu-Niculescu-Mizil, C. & Lee, L. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from wikipedia. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)* (pp. 365-368), Los Angeles, California, USA, Jun.

ELNAZ DAVOODI
DEPARTMENT OF COMPUTER SCIENCE
AND SOFTWARE ENGINEERING,
CONCORDIA UNIVERSITY, MONTREAL, CANADA.
E-MAIL: <E_DAVOO @ENCS.CONCORDIA.CA>

LEILA KOSSEIM
DEPARTMENT OF COMPUTER SCIENCE
AND SOFTWARE ENGINEERING,
CONCORDIA UNIVERSITY, MONTREAL, CANADA.
E-MAIL: <KOSSEIMG@ENCS.CONCORDIA.CA>

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 45–62
Received 19/02/2015, Accepted 15/05/2015, Final 20/06/2015.
ISSN 0976-0962, http://ijcla.bahripublications.com

Computational Study of Stylistics: A Clustering-based Interestingness Measure for Extracting Relevant Syntactic Patterns

MOHAMED-AMINE BOUKHALED
FRANCESCA FRONTINI
GAUVAIN BOURGNE
JEAN-GABRIEL GANASCIA
Université Pierre et Marie Curie and CNRS (UMR7606), France

ABSTRACT

In this contribution, we present a computational stylistic study of the French classic literature texts based on a data-driven approach where discovering interesting linguistic patterns is done without any prior knowledge. We propose an objective interestingness measure to extract meaningful stylistic syntactic patterns from a given author's work. Our hypothesis is based on the fact that the most characterising linguistic patterns should significantly reflect the author's stylistic choice in that the positions of theirs occurrences are controlled by the author's purpose, while the irrelevant linguistic patterns are distributed randomly in the text. Since it does not rely on the counts of occurrences of the syntactic patterns in texts, this measure can work reasonably well with both large and small text samples. The analysed results show the effectiveness in extracting interesting syntactic patterns from a single text, and this seems particularly promising for the analyses of such texts that, for their characteristics or for historical reasons, cannot support a comparative study.

Keywords: Computational stylistics, interestingness measure, sequential pattern mining, syntactic style

1. INTRODUCTION

Computational stylistic is the subfield of computational linguistics that aims to extract patterns of style characterizing a particular type of texts using some computational and automatic methods [5]. Addressing questions of style, it shares many commonalities with computational authorship attribution [18] in which one assign a text of unknown authorship to one of some candidate authors based on the stylistic information extracted from documents written by them. Rather than concentrating on those subconscious traits that may constitute an author's fingerprint, computational stylistics seeks to study those features of an author's style that are not only distinctive but also intentionally used by the author. Computational stylistic interferes in many other related tasks such as stylistic text classification [8], stylistic-based text generation [7] automatic readability and complexity assessment [12]. However the field of research in which the notion of style find its strong arguments is the computer-assisted literary analysis and computer-based literary criticism. Stylometric techniques have been used for nearly sixty years to study questions relating style (see [17] for a discussion and overview). First works focused more on lexical traits such as word counts, later more complex grammatical traits have been taken into account.

From the methodological point of view, two different types of approach have emerged:

- Classification approaches, that can be simplified as such: an a priori classification is found in literature (such as Shakespeare's comedies vs tragedies); some relevant linguistic features are identified and counted (such as function words) and finally clustering techniques are used to see whether the a priori distinction holds or not [5].
- Hermeneutic approach, in which texts are analysed in order to automatically extract significant features that may later be used by domain experts to produce a better informed and data driven critical analysis of texts [9, 14].

What such approaches often have in common is the fact that the distinctiveness of a text or of some of its elements is measured by comparison to other texts. The present work follows the hermeneutic approach, in that it seeks to extract significant syntactic patterns. However, the proposed methodology is based only on intrinsic evaluation, so that it can be applied to one text at a time.

In our contribution, we present a computational stylistic study of the French classic literature texts based on a data-driven approach where discovering interesting linguistic patterns is done without any prior knowledge.

We propose an objective interestingness measure to assist linguists in studying the syntactic style and in extracting meaningful linguistic patterns from a given author's work. More specifically, this interestingness measure is based on the position in which a pattern appears in the text, rather than its frequency. It is meant to support stylistic textual analysis by:

1. Verifying the degree of importance of each linguistic pattern (syntagmatic segments with gaps) via a new clustering-based interestingness measure.
2. Automatically inducing a list of linguistic features that are significant, representative for an author's work.

Thus, the goal of this paper is to present a practical intrinsic measure for extracting syntactic patterns from texts for stylistic analysis. This measure is motivated by statistical and linguistic considerations. Since it does not rely on the occurrences' counts of the syntactic patterns in texts, this measure can work reasonably well with both large and small text samples and allows the extraction of significant syntactic patterns.

The rest of the paper is structured as follow. We first give a linguistic motivation to our contribution with a brief overview of the methodology in Section 2. Then, in Section 3 we quantitatively analyze the syntactic order to show that the positions of the syntactic forms in the text are more relevant than their frequencies. We present the statistical formulation of the proposed intrinsic interestingness measure in Section 4. Finally a

thorough analysis and discussion of the patterns extracted with this methodology will be given in Section 5 for a selected classic French novel in order to illustrate the kind of stylistic patterns that can be identified.

2. MOTIVATION

In our study, we consider a syntagmatic approach. The text is first segmented into a set of sentences, and then each sentence is mapped into a sequence of syntactic (part of speech or POS-tag) items. For example, the sentence "Le silence profond régnait nuit et jour dans la maison" is first mapped to a sequence of:

$$< DET, NOM, ADJ, VER, NOM, KON, NOM, PRP, DET, NOM, SENT >!$$

Then sequential patterns of a determined length are extracted, such as:

$$< DET >< NOM >< ADJ >$$
$$< NOM >< ADJ >< VER >< NOM >$$
$$< KON >< NOM >< * >< DET >< NOM >$$

However, as sequential pattern mining is known to produce a large quantity of patterns even from relatively small samples of texts, an interestingness measure should be applied on these patterns in order to identify the most important ones. To the best of our knowledge, all the interestingness measures proposed in the literature to deal with this issue are based on the support of the pattern, that is, the frequency of occurrence of those patterns in the texts. However, frequency based methods are argued not to be precise in studying linguistic phenomena unless a huge quantity of texts is used.

Our hypothesis is based on the fact that the most characterizing linguistic patterns should significantly reflect the author's stylistic choice which makes theirs positions of occurrences controlled by the author's purpose, while the irrelevant linguistic patterns are distributed randomly in the text. Following this idea, the assumption made in this approach is that

the higher the importance of a linguistic pattern, the more its occurrences cluster together detaching themselves from a random distribution. By this methodology we search for patterns whose frequency is much higher in single portions of texts than in others, thus making each of them the locally most prominent pattern. The clustering phenomenon can be visualized in Figure 1 where we have plotted the absolute positions in the text of two different syntactical patterns. One can clearly notice that despite the two patterns having the same support (counts of sentences where they appear); they significantly behave differently in terms of their distribution of occurrences' positions. This property gives them a different linguistic relevancy value.

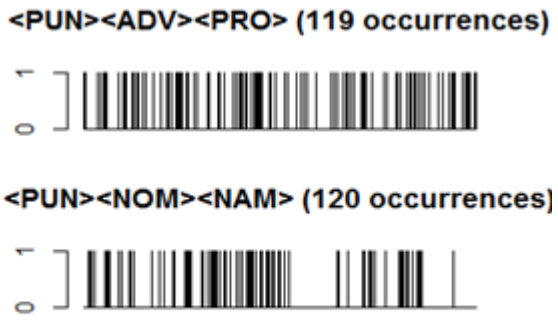


Figure 1. *Positions of occurrences in the text, counted by sentences from the first to the last one, of two different patterns with approximately same support, but with different distribution of positions. (A very thin vertical line is drawn at the position of each occurrence).*

3. QUANTITATIVE ANALYSIS OF THE SYNTACTIC ORDER

The written text is a very syntactically regulated phenomenon in the sense that not all the syntactical combinations are allowed to construct a well-formed syntactic sequence that can carry a semantic meaning.

There are two main factors acting at two different levels that regulate the syntactic order of a text. The first one is the grammar that acts on the phrase level by restricting the syntactic variations via a set of syntactic rules. These syntactic rules forbid certain syntactic sequences that are considered invalid, and allow other valid ones (syntactic sequence that respects these rules). The second element is the genre of the text which acts at the sentence level. In fact it is clear that a text written to be a poem will significantly differ from a text written to be a novel in terms of the syntactic forms that are incorporated on each ones of these two different types of genre. This is due to the linguistic constraints imposed by the rhetoric of the genre. These two elements will introduce a certain statistical order into the syntactic sequences.

Much of the works done so far to extract linguistic patterns - including syntactic patterns – from text is based on some statistical methods that relay mainly on the frequency of these linguistic patterns in that text. This means that the importance and significance of the patterns are evaluated according to its counts in the texts. In fact, this turns out to be not always true. For example, let's consider that we want to extract relevant syntactic patterns that are representative of the style of an author in a classic French novel. If we consider the part of speech 3-gram (simple syntactic pattern) as a minimum unit of the syntactic structure, we find that the statistical hierarchy of this unit can be represented by Zipf's law (see Figure 2). The Zipf low [20] states that the frequency of any unit is inversely proportional to its rank. Thus a frequent unit will occur approximately times as often as the direct following less frequent unit where is a constant that can be empirically deduced. The Zipf low, as known in statistics, has tendency to decrease the significance of the low frequency events, even if they are implicitly relevant to the studied subject. This applies also in the context of linguistics [10].

The property of the Zipf low, which makes the POS 3-gram behave in such a way on the one hand, and the statistical fluctuation on the other hand, increase the weaknesses of the methods based on the frequency as main element to evaluate the

relevancy. In fact, the frequency properties and their application have been discussed in the computational linguistic community. As a result, the frequency-based methods were claimed to be an unsophisticated way to discriminate the relevant linguistic forms in many cases, especially for the case of extracting relevant words from texts [15]. For instances, mutual information estimates based directly on counts are subject to overestimation when the counts involved are relatively small, and z-scores method that are based on the normality assumption substantially overestimate the significance of rare events [6].

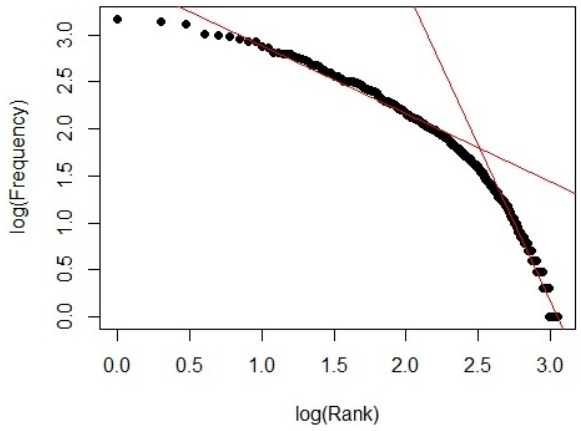


Figure 2. *A plot of POS 3-gram frequency in Hugo's novel "Notre-Dame de Paris". The plot is in log-log coordinates. x is rank of a word in the frequency table; y is the total number of the 3-gram's occurrences*

4. PROPOSED METHODOLOGY

Our method is based on two steps. First a sequential pattern mining algorithm is used to extract sequential syntactic patterns from the text. Secondly, the proposed interestingness measure is applied on the extracted set of syntactic patterns in order to identify the most relevant ones. Section 4.1 introduces some elements about the sequential syntactic pattern extraction

process. Then, the formulation and the statistical details of the proposed interestingness measure are presented in Section 4.2.

4.1. *Extracting sequential syntactic pattern*

Sequential data mining is a data mining subdomain introduced by [1] which is concerned with finding interesting characteristics and patterns in sequential databases, such as biological databases and customer databases in which data is mined to extract behavior sequential patterns. In what follow, for the sake of clarity, we will limit our definitions and annotations to those necessary to understand our experiment.

Table 1. *Sequence database SDB*

Sequence ID	Sequence
1	< a, b, d, e >
2	< a, b, c, e >
3	< b, d, e >

Let's consider a set of literals called items, denoted by $I = \{i_1, ..., i_n\}$. A sequence S (single-item sequence) is an ordered list of items, denoted by $S = <i_1...i_n>$ where $i_1...i_n$ are items. A sequence database *SDB* is a set of tuples *(id, S),* where *id* is the sequence identifier and S a sequence. Interesting characteristics can be extracted from such databases using sequential rules and pattern mining.

We define the sequential pattern as an ordered subset of I formed by the sequence combination of k recurring elements of the set *I.* The sequential pattern admits also the presence of one or many variable items denoted by "*" that can be substituted by any other items. Several algorithms have been developed to efficiently extract this type of patterns [19]. For example, if we run this algorithm on the *SDB* containing the three sequences presented in Table 1, we will get as a result sequential patterns, such us: "$< a >< b >$" with support equal to 2, which means that this pattern is respected by two sequences in the *SDB* (i.e., there exist two sequences of the *SDB* where we find the item *a*, we also find *b* just afterward in the same sequence), or:

"$< b > < * > < e >$" with support equal to 3, which means that this pattern is respected by tree sequences in the *SDB* (i.e., in all the sequences of the *SDB* one can find the item b followed by any item, followed by e).

Quiniou et al. [13] in their study have shown the interest of using sequential data mining methods for the stylistic analysis of large texts. They have shown that relevant and understandable patterns that are characteristic of a specific type of text can be extracted using sequential data mining techniques such as sequential pattern mining. They have considered the text as a set of sentences and each sentence as a sequence of ordered syntactic (POS tags1) or lexical (lemma) items using TreeTagger [16]. Each item in the sequence corresponds to one token (word) in the sentence respecting the order. Using this configuration as input for the sequential pattern mining algorithm, they pointed out that their method is better than machine learning methods such as Hidden Markov Models or Conditional Random Fields, in the sense that it produces outputs that are more understandable by humans. In our study, we will consider a similar configuration to that used by Quiniou et al. The text is first segmented into a set of sentences, then each sentence is mapped into a sequence of syntactic items (POS tags[1]). The whole text will produce a sequential database. Then, sequential patterns of a determined length are extracted from this syntactic sequential database using the sequential pattern algorithm. To avoid the effect of statistical fluctuation on the patterns with small support, we set the minimum support threshold constraint to 1%, that is, we focus only on patterns that are presents in at least 1% of the sentences of the analysed text. Example about syntactic patterns are presented and discussed in Section 5.

4.2. *Filtering the most relevant sequential syntactic pattern*

The positions of the pattern occurrences with support equal to in the ordered set of sentences of the text are denoted by $P = (p_1, p_2, p_3, ...p_n)$ where p_1 is the rank of the sentence in which

[1] POS tag list for French: <www.cis.uni-muenchen.de/~schmid/tools/ TreeTagger/data /french-tagset.html>.

the pattern appears for the -th time $i \in (1,..,n)$ with. The set of distances between two successive occurrences of a pattern can be denoted by $D = (d_1, d_2, d_3,...,d_{n-1})$ where $di = p_{i+1} - p_1$.

Given that configuration, in order to quantify the degree of importance of each linguistic pattern and thus evaluate its relevance based on its clustering behaviour, we use the parameter which is defined as the standard deviation of the distribution of the set of distances normalised by the expected value of such as:

$$\sigma = \frac{\sqrt{E(D^2) - E(D)^2}}{E(D)} \quad (1)$$

For the case where the distribution of the pattern position set P is completely random, one should expect the corresponding distances set D to follow a Geometric distribution and thus to have a parameter $\sigma = 1$, and the larger is the clustering the bigger is σ [11]. However, patterns with different random distributions of D (different random clustering settings in a text) would have a significant difference in their corresponding parameter σ. To avoid this, we normalise the pattern parameter σ with the Geometric parameter σ_{geo} where $\sigma_{geo} = \sqrt{1 - (1/E(d))}$ such as:

$$\sigma_{norm} = \frac{\sigma}{\sigma_{geo}} \quad (2)$$

Such method was already successfully used in physics to quantify energy level of disordered system and in information retrieval to extract key words and keys phrase from informative texts [4].

5. Experimental Validation and Discussion

In this section, some of the significant patterns extracted and ranked in terms of relevancy with the proposed method from Victor Hugo's novel "*Notre-Dame de Paris*" (NDdP) are described and discussed.

In order to appreciate the results, we shall compare them with the mere study of frequent patterns (frequency as interestingness measure). Frequency based approaches may be used in comparative works, as in such cases it is possible to filter out patterns that are frequent in all texts and thus uninteresting. In an intrinsic approach this is not possible, thus the most frequent patterns are not very informative per se:

For example, the Pattern (1): $< VER > < DET > < NOM >$ (support = 3101) is one of the most frequent ones in NDdP. By comparing NDdP with other novels by contemporary authors, we might find out that Hugo under/over uses this syntactic structure, and possibly draw some conclusions. But in isolation this doesn't tell us much, as it is clear that the verb-object structure is very common in French.

The same is true for the Pattern (2): $< NOM > < PRP > < NOM >$ (support = 3101) that presents a simple noun phrase modified by a prepositional phrase which is a frequent structure in French.

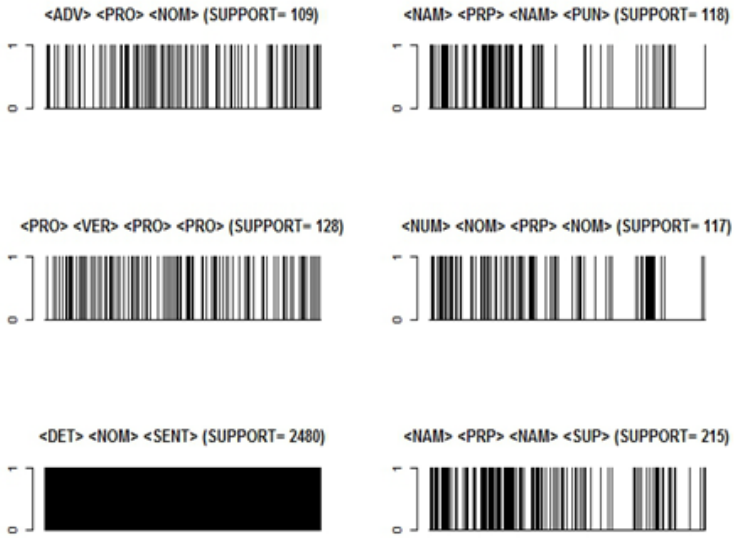


Figure 3. *Positions plot of three relevant syntactic patterns (right column) versus three non-relevant patterns (left column)*

On the contrary, by using the proposed method the extracted patterns seem to bear a strong relation to this particular text, its story line and the literary genre it instantiates, namely that of the historical novel.

Let us here take into account some examples:

- Pattern (3): *< NOM > < PRP > < NAM > < PUN >* support = 340 instances in the text:

  — …tour de Notre-Dame ,
  — …hôtel de Bourbon ,
  — …murailles de Paris ,
  — …prince de Conty :
  — …dauphin de Vienne ;
  — …échevin de Bruxelles ;

In Pattern (3) the proper name is often a location, especially at the beginning of the novel where descriptive parts are more frequent for the purpose of guiding the reader into the topography of medieval Paris. Later it serves the purpose of locating the plot. Other instances of this pattern are used to mention characters, especially historical ones, by their title and provenance. This also is very typical of a literary genre where historical elements are combined with fictional ones.

The skewedness of the distribution of this pattern is also due to the fact that some of the parts of the novel are more descriptive; they serve the purpose of introducing the historical setting and the characters, while other parts develop the action and thus do not introduce many new characters. Other extracted patterns have a similar function.

Pattern (4) is often used to introduce characters by first stating their name and title. It is worth noticing that NDdP presents a plethora of minor characters (see positions plot of this pattern in Figure 3):

- Pattern (4): *< NAM > < PRP > < NAM > < PUN >* support= 118, instances in the text:

- …Marguerite de Flandre,
- …Jehan de Troyes ,
- …Frollo du Moulin ,

The same is true for Pattern (5) which is also used to introduce character's full names (patronymics).

- Pattern (5): *< PUN > < NOM > < NAM >* support = 120

Pattern (6) instead is often instantiated in presentative structures such as (6.A and 6.B), with the topic (here the person) in focus position, used for changes of scenes, to introduce new characters.

In the final part of NDdP, this pattern instantiates other kinds of structures, such as (6.C), which are used to represents actions.

- Pattern (6) : *< PRP > < VER > < NAM >*, support = 113

  - (6.A) Il y avait pourtant une créature humaine que Quasimodo exceptait de sa malice et de sa haine pour les autres, et qu'il aimait autant, plus peut-être que sa cathédrale; ***c'était Claude*** Frollo. [it was Claude]
  - (6.B) ***C' était Quasimodo***, sanglé, cerclé, ficelé, garrotté et sous bonne garde. [it was Quasimodo]
  - (6.C) …acculés à Notre-Dame qu' ils assaillaient encore et ***que défendait Quasimodo***,… [that Quasimodo was defending]

As we have seen, many of the significant patterns extracted with this technique contain the NAM (proper name) tag. This happens not only with sequential pattern mining, but also with other statistical pattern mining methods, as in general proper names are less frequent than other tags and their skewed distribution causes them to emerge in significance measurements. In a study that focuses purely on syntax it may be worth merging this class with the one of common names.

Pattern (7) does not contain proper names, and seems very relevant for the text in question. Among the instances of this

pattern we find many vivid and precise descriptions, as is evident especially in (7.B) where Hugo lists all the different divisions that used to be in charge of the defence of the former stronghold of "Châtelet", in Paris.

- Pattern (7): *< NUM > < NOM > < PRP > < NOM >* support = 117 , instances in the text:

  - (7.A) …le fracas de tous les gros doubles pétards de la Saint-Jean, la décharge de ***vingt arquebuses à croc***, la détonation de cette fameuse serpentine de la Tour de Billy, [twenty arquebuses]
  - (7.B) …les cent ***vingt sergents à cheval***, les cent ***vingt sergents à verge***, le chevalier du guet avec son guet, son sous-guet, son contre-guet et son arrière-guet? [twenty seargents on horse]

The few analysed examples indicate that the presented technique is effective in extracting interesting syntactic patterns from a single text, and this seems particularly promising for the analyses of such texts that, for their characteristics or for historical reasons, cannot support a comparative study as they are, in some way, unique. This might be the case of great poems from the antiquity, such as the Iliad or the Odyssey or even contemporary works whose style is too peculiar for comparison, such as James Joyces's Ulysses.

On the other hand, this technique, as well as other similar ones, prompts the question of what is really captured by significant patterns. Some structures may be significant because they are typical of an author's style, its fingerprint – as we may say borrowing a metaphor often used in attribution studies, or they may be dictated by functional needs, due to the particular topic of the work, or to the conventions of the chosen genre. This is particularly true for syntactic analysis, where the functional constraints on the authorial freedom are more evident.

Using Biber & Conrad [3] definitions, it is worth asking how far it is possible to distinguish style from register and genre when

analysing syntactic structure, especially considering that stylometric studies have given their best results with features such as word and sentence length, or lexical richness. In particular this technique, based on the uneven distribution of patterns, will invariably capture local changes in register motivated by the different elements of the novel (introduction, descriptions, scenes of action, dialogue, ...), alongside with stylistic traits.

It is always hard in linguistics to separate the form from the function. For this reason it is important to study syntactic patterns in the light of the sentences from which they are drawn to avoid false conclusions. Nevertheless the technique seems efficient in demoting those frequent constructions that are typical of French syntax in general without the need of a reference corpus; at the same time the syntactic structure of the extracted patterns and their use in vivid descriptions, in the presentation of characters and in the reconstruction of scenes do seem to resonate with the particular use of language typical of Victor Hugo's prose in NDdP.

6. CONCLUSION

In this paper, we have presented an objective interestingness measure to extract meaningful stylistic syntactic patterns from a given author's work. Our hypothesis is based on the fact that the most characterizing linguistic patterns should significantly adopt a clustering behaviour which detaches them from a random positioning in the text. To evaluate the effectiveness of the proposed method, we conducted an experiment on a classic French Novel. The analysed results show the effectiveness in extracting interesting syntactic patterns from a single text.

Based on the current study, we have identified several future research directions. First, we will explore combining the support with the distribution to calculate the interestingness measure. Second, this study will be expanded to include morpho-syntactic patterns. Third, we intend to experiment with other languages and text sizes using standard corpora employed in the field of computational stylistics at large.

REFERENCES

1. Agrawal, R., Imieli'nski, T. & Swami, A. 1993. Mining association rules between sets of items in large databases. In *ACM SIGMOD Record* (pp. 207-216).
2. Biber, D. 2006. *University Language: A Corpus-based Study of Spoken and Written Registers*. John Benjamins Publishing.
3. Biber, D. & Conrad, S. 2009. *Register, Genre, and Style*. Cambridge University Press.
4. Carpena, P. et al. 2009. Level statistics of words: Finding keywords in literary texts and symbolic sequences. *Physical Review E*, 79/3, 35102.
5. Craig, H. 2004. Stylistic analysis and authorship studies. *A Companion to Digital Humanities*, 3, 233-334.
6. Dunning, T. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19/1, 61-74.
7. Hovy, E. H. 1990. Pragmatics and natural language generation. *Artificial Intelligence*, 43/2, 153-197.
8. Kessler, B., Numberg, G. & Schütze, H., 1997. Automatic detection of text genre. In proceedings of the *35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics* (pp. 32-38).
9. Mahlberg, M. 2012. *Corpus Stylistics and Dickens's Fiction*. Routledge.
10. Montemurro, M. A. 2001. Beyond the Zipf-Mandelbrot law in quantitative linguistics. *Physica A: Statistical Mechanics and its Applications*, 300/3, 567-578.
11. Ortuno, M. et al. 2002. Keyword detection in natural languages and DNA. *EPL (Europhysics Letters)*, 57/5, 759.
12. Pitler, E. & Nenkova, A. 2008. Revisiting readability: A unified framework for predicting text quality. In proceedings of the *Conference on Empirical Methods in Natural Language Processing* (pp. 186-195).
13. Quiniou, S. et al. 2012. What about sequential data mining techniques to identify linguistic patterns for stylistics? In *Computational Linguistics and Intelligent Text Processing* (pp. 166-177). Springer.

14. Ramsay, S. 2011. *Reading Machines: Toward an Algorithmic Criticism*. University of Illinois Press.
15. Salton, G. & McGill, M. J. 1983. Introduction to modern information retrieval.
16. Schmid, H. 1994. Probabilistic part-of-speech tagging using decision trees. In proceedings of the *International Conference on New Methods in Language Processing* (pp. 44-49).
17. Siemens, R. & Schreibman, S. 2013. *A Companion to Digital Literary Studies*. John Wiley & Sons.
18. Stamatatos, E. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60/3, 538-556.
19. Viger, P. F. et al. 2014. SPMF: A java open-source pattern mining library. *Journal of Machine Learning Research*, 15, 3389-3393.
20. Zipf, G. K. 1949. Human behavior and the principle of least effort.

ACKNOWLEDGMENT

This work was supported by French state funds managed by the ANR within the Investissements d'Avenir programme under reference ANR-11-IDEX-0004-02.

MOHAMED-AMINE BOUKHALED
LIP6 (LABORATOIRE D'INFORMATIQUE DE PARIS 6),
UNIVERSITÉ PIERRE ET MARIE CURIE AND CNRS (UMR7606),
ACASA TEAM, 4, PLACE JUSSIEU,
75252-PARIS CEDEX 05, FRANCE.
E-MAIL: <MOHAMED.BOUKHALED@LIP6.FR>

FRANCESCA FRONTINI
LIP6 (LABORATOIRE D'INFORMATIQUE DE PARIS 6),
UNIVERSITÉ PIERRE ET MARIE CURIE AND CNRS (UMR7606),
ACASA TEAM, 4, PLACE JUSSIEU,
75252-PARIS CEDEX 05, FRANCE.
E-MAIL: <FRANCESCA.FRONTINI@LIP6.FR>

GAUVAIN BOURGNE
LIP6 (LABORATOIRE D'INFORMATIQUE DE PARIS 6),
UNIVERSITÉ PIERRE ET MARIE CURIE AND CNRS (UMR7606),
ACASA TEAM, 4, PLACE JUSSIEU,
75252-PARIS CEDEX 05, FRANCE.
E-MAIL: <GAUVAIN.BOURGNE@LIP6.FR>

JEAN-GABRIEL GANASCIA
LIP6 (LABORATOIRE D'INFORMATIQUE DE PARIS 6),
UNIVERSITÉ PIERRE ET MARIE CURIE AND CNRS (UMR7606),
ACASA TEAM, 4, PLACE JUSSIEU,
75252-PARIS CEDEX 05, FRANCE.
E-MAIL: <JEAN-GABRIEL.GANASCIA@LIP6.FR>

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 63–79
Received 26/02/2015, Accepted 21/05/2015, Final 16/06/2015.
ISSN 0976-0962, http://ijcla.bahripublications.com

What do our Children Read About? Affect Analysis of Chilean School Texts

CLAUDIA MARTÍNEZ
JORGE FERNÁNDEZ
Universidad Católica de la Santísima Concepción, Chile

ALEJANDRA SEGURA
CHRISTIAN VIDAL-CASTRO
CLEMENTE RUBIO-MANZANO
Universidad del Bío-Bío, UBB, Concepción, Chile

ABSTRACT

We present a study of the affective character of 1st to 8th year Chilean school texts, to which we applied lexicon-based affect analysis techniques to identify 6 basic emotions (anger, sadness, fear, disgust, surprise and happiness). First, we generated a corpus of 525 documents, 18176 paragraphs and 137516 words. Then, using the affective words frequency, we built a classifier based on Emotion Word Density to detect emotions in the texts. Our results show that the predominant affective states are happiness (58%), sadness (16%) and fear (12%). The 6 basic emotions are present in most literary forms with uniform relative density except for songs, where anger is absent. Classifier performance was validated by comparing its results against the opinions of experts in the field, and its results show an above-average conformity (accuracy = 63%), above-average predictive capacity (precision = 69%) and good classifier sensitivity (recall = 80% and f-measure = 93%).

Keywords: Affect analysis, sentiment analysis, subjectivity analysis

1. INTRODUCTION

Affect analysis is a branch of natural language processing focused on the development and application of techniques that estimate the emotive aspects of a text [1]. Several psychological theories commonly used in affect analysis classify emotions into between six and nine basic classes. For example, Ekman [2] defines the six basic emotions as anger, disgust, fear, joy, sadness and surprise. In [3], Plutchik and Kellerman add guilt, interest and shame to the basic emotion classes. These approaches assume that all other emotions can be assigned to these basic groups, on which other theories and classifications are based. This study aims to detect emotions in school texts provided by the Chilean Education Ministry for 2014 based on a lexical resource.

The rest of the article is organized as follows: in the next section, it reviews the relevant bibliography about lexicon-based affect analysis techniques. The next two sections describe the various techniques, tools and metrics used in this study. Experiments and their results are described in sections 5 and 6, while section 7 and 8 present the expert validation stage and a discussion of the results.

2. BACKGROUND

Affect analysis is usually applied using an approach based on machine learning, where automated or supervised learning algorithms detect and classify texts into affective categories. Another common approach uses lexical structures, specifically lexicon containing sets of words categorized according to their affective content [4], [5].

Affect analysis and sentiment analysis have been applied to literary forms to gather information about the emotions that these texts might provoke the reader. Most works that have applied the lexicon-based strategy to this purpose have taken advantage of the availability of English-language lexicons. For example, Neviarouskaya in [6] applied the ATtitude Analysis Model to the classification of fairy tales using the Izard's theory of 9 emotions

[7] to detect attitudes, and using polarity analysis to detect judgments and appreciations. Later, Mohammad [8] proposed a technique for visualizing emotion-related words based upon the frequency of the emotions. This study does not focus on the detection of emotions themselves, but rather the visualization of the presence of emotionally-loaded words in certain children's stories, specifically 192 fairy tales from the Brothers Grimm. Similarly, Ptaszyinki et al. present a similar approach for the Japanese language in [9], where a similarity metric was used to expand a rather small group of emotionally-charged words (containing 503 nouns) into an emotions dictionary (containing 15612 verbs), which was later used in the syntactic rules applied to the Snow White story. The study´s results were not encouraging mainly due to dictionary quality and syntactical rules incoherence. In [10], Sugimoto and Masahide experimented with an automatic reading system that was able to read aloud texts and novels with emotional emphasis. Texts were classified according to their emotion word distribution. At the same time, a sentence's emotion was classified according to the emotions associated to its nouns, adjectives and/or verbs. In [11], Ptaszynski et al. propose a method for extracting emotion information from narrative texts by analyzing anaphoric expressions, from which the emotional state of each character in the each stage of the narrative can be determined. Finally, [12] compares studies about the use of lexicons for affect analysis in tweets.

In this study, in contrast to previous studies, school texts in Spanish are analyzed by applying the metric proposed by Mohammad [8], using an affective lexicon also in Spanish.

3. METHODOLOGY

Figure 1 presents this study´s work methodology, from text retrieval to the gathering of results and their evaluation.

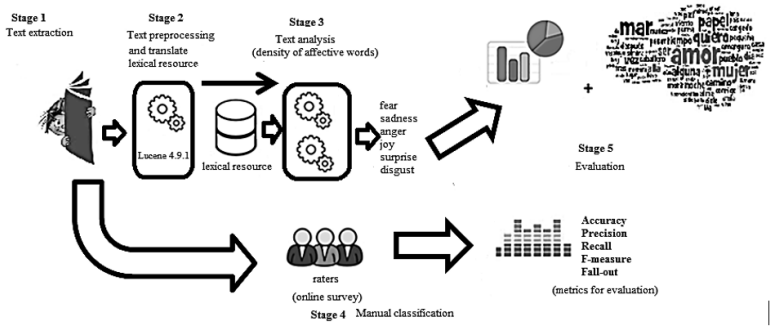


Figure 1. *Study methodology*

The methodology included four stages; the first stage was an exploratory search of official 2013 Ministry of Education texts published by Santillana Editions, next we extracted and classified the school texts from 1st to 8th grades into poems, stories, novels, songs, articles, news, interviews, and biographies. In the third stage, texts were preprocessed using the API Lucene 4.9.1, removing stopwords, whitespaces and special characters; this stage ends with the application of the SnowBall stemming algorithm. Next, an expert translated the lexical resource from English to Spanish and then detected the affective words frequency based on the emotion word density metric [8]; at the same time, in the fourth stage, it applied a survey to two raters so as to do a comparison between manual and automatic classification.

4. DEVELOPMENT

4.1. *Text extraction stage*

Texts for this study were obtained from the catalog of school texts provided free of charge by the Chilean Ministry of Education to students and teachers of all public and publicly-subsidized schools in Chile. In particular, texts were extracted from the 1st year to 8th year Language and Communication textbooks published by Editorial Santillana. These texts are categorized into 15 literary forms: Story, Reading, Poem,

Dramatic work, Fable, Legend, Myth, Article, Novel fragment, Interview, News report, News item, Biography and Other. We generated a corpus of 525 documents, 18176 paragraphs and 137516 words.

4.2. *Text preprocessing stage*

In this phase, texts were cleaned and prepared for the classification phase, so as to improve the classifier´s performance and speed up the classification process, thus aiding the affect analysis process. This phase includes several transformation stages: online text cleaning, white space removal, abbreviation expansion, stopwords removal and stemming. Finally, the preprocessed text undergoes a filtering stage, during which several pattern selection functions are applied [13].

Text cleaning involved removing any characters no considered useful for this study (stopwords, white space, special characters, etc.). The unit of analysis chosen for this work is the paragraph. Thus, periods were not removed as they act as paragraph separators. Both lexicon words and text words were stemmed prior to the matching phase. The Apache Lucene[1] 4.9.1 API was used, which includes the following modules:

a. **StopAnalyzer**: This analyzer is used to remove articles, pronouns, conjunctions, etc., as they are considered irrelevant for the purposes of this study.
b. **WhitespaceAnalyzer**: This analyzer is used to remove extra whitespace, yielding a text free of extra whitespace and concatenated words.
c. **SimpleAnalyzer**: This analyzer is used to transform all text to lowercase and to eliminate all punctuation marks, yielding a text that is case consistent.
d. **SnowballAnalyzer**: This analyzer is used to stem both the text words and the lexicon using the Snowball algorithm.

R-Project package version 3.0.1 was used in the final preprocessing stage was done using, in particular its RCurl, Rjson

[1] https://lucene.apache.org/core/

and TM packages were utilized for corpus generation and preprocessing. Likewise, the Rcpp, RColorBrewer andWordcloud R-Project packages were used to generate frequency diagrams and word clouds. Based on Hassan-Montero's work [14], we chose to use tagcloud visualization techniques to represent the most frequently occurring words in the texts.

An emotion dictionary based on WordNetAffect was used as the lexical resource [15]. As this dictionary was available only in English, it was translated into Spanish by an expert. This resource was built by selecting a subset of synsets suitable for representing affective concepts; additional information was also added to the affective synsets without defining new ones. We kept the structure of the original lexicon, thus generating a set of infrequently used words in Spanish for doing affective analysis. This dictionary includes a total of 1523 words (nouns, adjectives and verbs) organized into 6 emotion classes: anger, disgust, fear, sadness, joy and surprise. Additionally, some synonyms were added to expand the knowledge base. Table 1 shows that, after the preprocessing stage the corpus was reduced in size from 799033 words to only 137516 words.

Table 1. *Number of words per literary form*

Literary form	Including stopwords	Not including stopwords	Percentage of total
Story	147720	24737	18%
Reading	396126	62641	46%
Poem	56811	9637	7%
Dramatic work	39103	6536	5%
Legend	26740	4339	3%
Myth	25653	4183	3%
Novel fragment	14327	2424	2%
Article	39302	5717	4%
Biography	5721	860	1%
Song	1633	891	1%
News report	10096	1546	1%
Interview	17061	5996	4%
News	4503	1171	1%
Fable	3545	1735	1%
Others	10692	5103	4%
Total	799033	137516	100%

The "Other" category includes texts that cannot be categorized as belonging to any other literary form.

4.3. *Text analysis stage (density of affective words)*
For this phase, we wrote a Java script to calculate the Emotion Word Density [8], which represents the relative percentage of a word over the universe of emotion words analyzed, as shown in Equation 1. Through this parameter, we can determine the frequency of emotions present in each literary form.

$$F(e_1,t_1) = \frac{count(e_1,t_1)}{N_1} \quad (1)$$

where $F(e_1, t_1)$ is the relative frequency of affect word e_1, given that $e_1 \in t_1 \wedge e_1 \in$ lexical resource R_1, $e_1 \in t_1$, $count(e_1\ t_1)$ is the occurrence of e_1 in text t_1 y N_1 the total volume of affect words found in text t_1. According to Mohammad [8], this analysis strategy might not be always trustworthy to determine if a given phrase is expressing a certain emotion, but can be used to determine whether a particular section of text has more expressions of emotion than other sections. As mentioned before, the school texts under study are categorized into 15 literary forms, ranging from narrative texts to argumentative texts, including descriptive texts and dialogues.

5. RESULTS

Figures 2 and 3 present the main results of the text analysis phase (automatic labeled). From Figure 2, it can be seen that the predominant emotion is joy (58%), followed by sadness (16%) and fear (12%). Also, Figure 3 shows that the six basic emotions are present in most literary forms with a uniform relative density except for the case of songs, where all emotions except anger are present. However, their absolute densities are very different, with the longer literary forms such as stories or reading having higher emotion word densities.

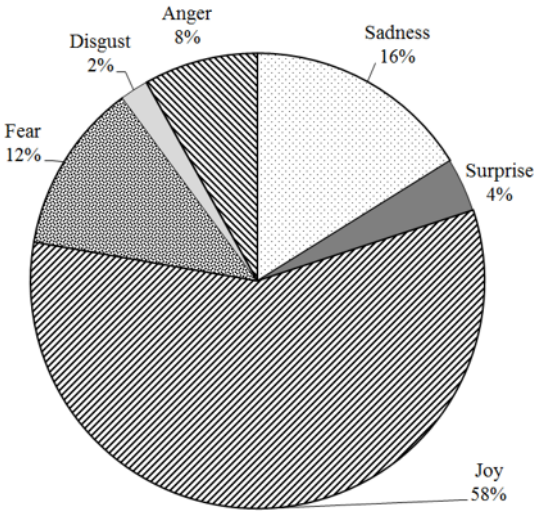


Figure 2. *Emotion density for the corpus*

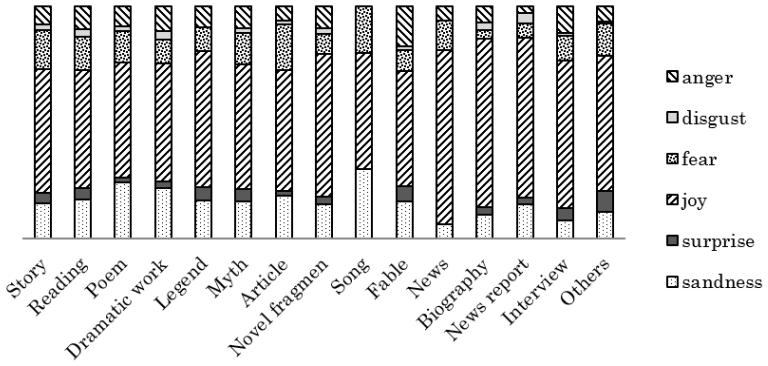


Figure 3. *Emotion relative density for each literary form*

As an example, we present detailed results for an instance of the Fable literary form: the 2nd year school text "The rainbow fish" in its Spanish translation. Figure 4 shows relative emotion densities as well as its corresponding tag cloud generated using the affect lexicon.

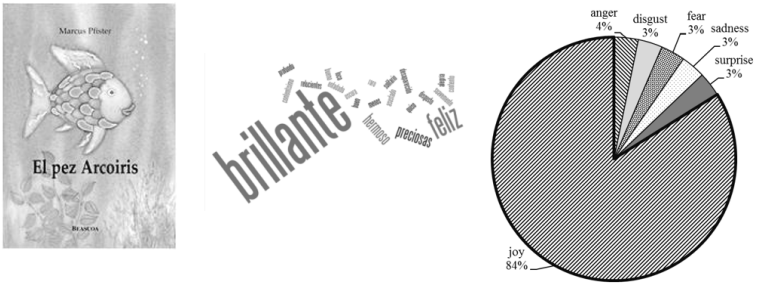


Figure 4. *Results for "El pez arcoiris", a text categorized in the Fable literary form*

In a further analysis phase, the 15 literary forms were clustered according to their size in words to aid the subsequent expert evaluation. Two homogeneous clusters were defined, comprising 54% and 46% of the words in the corpus, respectively. These clusters, as well as the literary forms and their respective sizes, are shown in Table 3.

Table 3. *Clustering of the literary forms*

	Literary form	**Number of documents**	**Number of words**	**Number of paragraphs**
Cluster 1	Story	55	24737	3435
	Poem	137	9637	1321
	Dramatic work	11	6536	909
	Legend	21	4339	333
	Myth	16	4193	322
	Article	28	39302	914
	Novel fragment	5	2424	333
	Fable	24	1735	13
	News item	5	1171	105
	Biography	7	5721	133
	News report	3	1546	235
	Interview	5	5996	461
	Song	17	891	58
	Other	28	5103	392
	Subtotal	362	74875	8964
Cluster 2	Reading	163	62641	9212
	Subtotal	163	62641	9212
	Total	525	137516	18176

Figure 5 presents relative emotion density results per cluster, where it can be seen that all six emotions are present in both clusters with uniform relative densities.

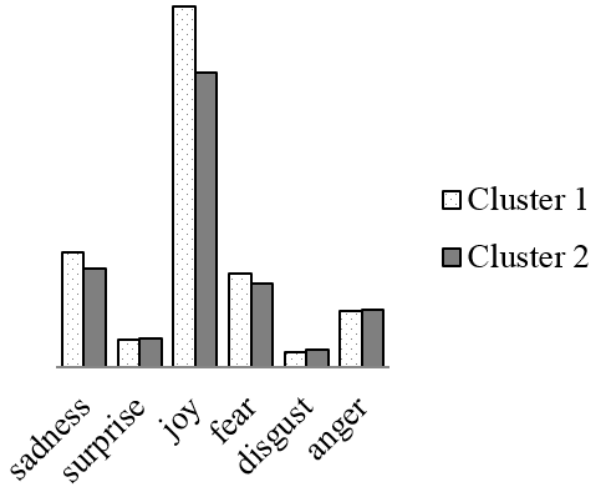
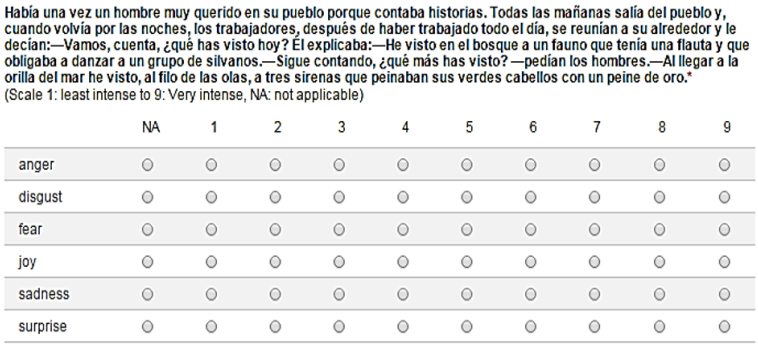


Figure 5. *Relative emotion density for each literary form cluster*

6. EVALUATION STAGE

The results of automatic classification were validated by gathering opinions and value judgments from experts in the field. To this end, it created 24 different surveys including 11 different paragraphs, for a total sample of 264 paragraphs. Each survey was designed so as to determine the emotional intensity of each paragraph regarding each of the six basic emotion classes. Assessment was done using the "Self-Assessment Manikin" (SAM) scale, containing nine categories ranging from least intense to very intense.



Había una vez un hombre muy querido en su pueblo porque contaba historias. Todas las mañanas salía del pueblo y, cuando volvía por las noches, los trabajadores, después de haber trabajado todo el día, se reunían a su alrededor y le decían:—Vamos, cuenta, ¿qué has visto hoy? Él explicaba:—He visto en el bosque a un fauno que tenía una flauta y que obligaba a danzar a un grupo de silvanos.—Sigue contando, ¿qué más has visto? —pedían los hombres.—Al llegar a la orilla del mar he visto, al filo de las olas, a tres sirenas que peinaban sus verdes cabellos con un peine de oro.*
(Scale 1: least intense to 9: Very intense, NA: not applicable)

	NA	1	2	3	4	5	6	7	8	9
anger	○	○	○	○	○	○	○	○	○	○
disgust	○	○	○	○	○	○	○	○	○	○
fear	○	○	○	○	○	○	○	○	○	○
joy	○	○	○	○	○	○	○	○	○	○
sadness	○	○	○	○	○	○	○	○	○	○
surprise	○	○	○	○	○	○	○	○	○	○

Figure 6. *Example survey used to evaluate a text's emotional intensity*

As shown in Figure 6, for each emotion class, the expert must mark the emotional intensity felt for the given text. Stratified proportional sampling was used to calculate the number of paragraphs to include in each survey category in accordance to population characteristics. Each kind of survey included paragraph samples from both clusters (48% of paragraphs from cluster 1 and 52% of paragraphs from cluster 2). Each one of the 24 kinds of surveys was answered by two experts. Given the complexity of the evaluation instrument, no inter-agreement analysis was performed at this research stage. Each expert evaluated all six emotion classes, some emotion classes share the polarity or generate similar emotions, additionally, the surveys included a 1 to 9 intensity scale. These characteristics also make the probability of concordance be very low.

The WordNetAffect graph structure classifies emotions using four different polarity levels: positive emotion, negative emotion, neutral emotion and ambiguous emotion. In this study we consider only two polarities: joy and surprise are considered positive emotions, while anger, disgust, sadness and fear are considered negative emotions.

We also take the following four premises into account:

Premise 1: We consider the survey results to be the truth.

Premise 2: If the survey and the automatic classifier match the emotion, we consider it a True Positive (TP) or True Negative (TN), correspondingly.
Premise 3: If the survey and the automatic classifier do not match the emotion but match the polarity, we also consider it a TP or TN, correspondingly.
Premise 4: If the survey and the automatic classifier do not match neither the emotion nor the polarity, we consider it a FP or FN, correspondingly.

Then, we discarded all cases for which the automatic classifier did not detect any emotions, and those cases for which automatic classification was ambiguous (same number of occurrences for 2 or more emotion classes).

Finally, we construct the analysis matrix for the 6 basic emotion classes shown in Table 4.

Table 4. *Analysis matrix for the 6 emotion classes*

	NEGATIVE				POSITIVE	
Automatic classifier / Manual classifier	angry	fear	disgust	sadness	surprise	joy
angry	TN	TN	TN	TN	FP	FP
fear	TN	TN	TN	TN	FP	FP
disgust	TN	TN	TN	TN	FP	FP
sadness	TN	TN	TN	TN	FP	FP
surprise	FN	FN	FN	FN	TP	TP
joy	FN	FN	FN	FN	TP	TP

The performance of the automatic classifier can be studied by calculating the metrics described in equations (2) to (6). These metrics differ from the classic information retrieval metrics: whereas the classic metrics are calculated using all retrieved and relevant documents into account, these modified metrics consider

only a representative sample of the paragraphs, classified according to their literary form.

$$ACCURACY' = \frac{TP + TN}{TP + FP + TN + FN} \quad (2)$$

$$PRECISION' = \frac{TP}{TP + FP} \quad (3)$$

$$RECALL' = \frac{TP}{TP+FN} \quad (4)$$

$$F - MEASURE' = \frac{2 * PRECISION' * RECALL'}{RECALL' + PRECISION'} \quad (5)$$

$$Fall - Out(FRP) = \frac{FP}{FP + TN} \quad (6)$$

Metrics results for the automatic emotion classifier are presented in Table 5.

Table 5. *Metrics results*

Metric	Value
Accuracy	0,63190184
Precision	0,69105691
Recall	0,79439252
F-measure	0,93043478
Fall-out **(**FRP)	0,67857142

7. RESULTS AND DISCUSSION

Comparisons between the automatic classifier's and survey results show an overall above-average conformity to the correct values (accuracy = 63%), above-average predictive capacity

(precision = 69%) and good classifier sensitivity (recall = 80% and f-measure = 93%). The True Positive rate was 80%, while the True Negative rate was only 32%. This indicates that the automatic classifier has an acceptable prediction rate for the detection of positive emotions (recall = 80%), but fails to predict negative emotions (VPR = 32%), thus giving a high false alarm rate (FPR = 68%). This in turn may be explained by the high number of negative sentences in the texts, such as "I will never be happy again", "The children were unable to show their love," etc. It must also be noted that the percentage of positive emotions is always higher than the percentage of negative emotions for all literary forms except songs, which can be considered as having neutral polarity. Finally, we must remark on the fact that of the 137516 words in the corpus, only 8250 of them were emotion-related, of which only 1821 were unique words.

8. Conclusions and Future Work

This study shows that the predominant emotion in Chilean school texts is happiness (68%). Whether intended or not, this was the expected result. Less expected was that the second and third most common emotions are sadness (16%) and fear (12%). The current lexicon's restrictions notwithstanding, we can state that is possible to predict emotion in Spanish texts using lexicon-based affect analysis techniques. However, the classifier's regular predictive capacity and conformity with the true values may be explained by considering that only 6% of the words in the Spanish-language texts belong to an emotion class. Also, the lexicon used in this work was created for the English language and it may well be that its translation into Spanish did not always capture correctly every phrase sense. One of the lessons learned through this work is that the construction of a lexicon for affect analysis in Spanish requires, rather than a translation, an interpretation of the relevant phrases and their synonyms so as to represent their real sense and their real usage in the target language. Likewise, we did not consider colloquial Chilean expressions in this work. In this regard, this work is only the first step in our research, and we are working on including negative

sentence handling and the use of amplifiers and simplifiers. We also will work on local ironies, so as to better understand phenomena related to subjectivity. Finally, given that many primary school level texts are usually short in length and use a limited vocabulary (only 8250 of the 137516 words can be found in the affective lexicon resource used), these results validate our approach and encourage us to pursue further research in this area.

REFERENCES

1. Grefenstette, G., Qu, Y., Shanahan, J. G. & Evans, D. A. 2004. Coupling niche browsers and affect analysis for an opinion mining application. *Proceedings of Recherche D'Information Assistée par Ordinateur (RAIO)*.
2. Ekman, P. 1993. Facial expression and emotion. *American Psychologist*, 48/4, 384-392.
3. Plutchik, R. & Kellerman, H. 1980. A general psychoevolutionary theory of emotion. In R. Plutchik & H. Kellerman (Eds.), *Emotion: Theory, Research, and Experience: Vol. 1. Theories of Emotion* (pp. 3-33). New York: Academic Press.
4. Strapparava, C. & Mihalcea, R. 2008. Learning to identify emotions in text. In proceedings of the *2008 ACM Symposium on Applied Computing* (pp. 1556-1560).
5. Balahur, A.; Mihalcea, R. & Montoyo, A. 2014. Computational approaches to subjectivity and sentiment analysis: Present and envisaged methods and applications. *Computer Speech & Language*, 28/1, 1-6.
6. Neviarouskaya, A., Prendiger, H. & Ishizuka, M. 2010. Recognition of affect, judgment, and appreciation in text. In proceedings of the *23rd International Conference on Computational Linguistics (COLING'10)* (pp. 806-814), Aug.
7. Izard, C. E. 1977. *Human Emotions*, New York: Plenum Press.
8. Mohammad, S. 2011. From *Once Upon a Time* to *Happily Ever After*: Tracking emotions in novels and fairy tales. In proceedings of the *ACL Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH)* (pp. 105-114), Portland, OR, Jun.
9. Ptaszynski, M., Rzepka, R., Araki, K. & Momouchi, Y. 2014. Automatically annotating a five-billion-word corpus of Japanese

blogs for sentiment and affect analysis. *Computer Speech & Language*, 28/1, 38-55.
10. Sugimoto F. & Masahide, Y. 2006. A method for classifying emotion of text based on emotional dictionaries for emotional reading. In proceedings of the *2nd International Symposium on Communications, Control and Signal Processing*, Marrakech, Morocco, Mar.
11. Ptaszynski, M., Dokoshi, H., Oyama, S., Rzepka, R., Kurihara, M., Araki, K. & Momouchi, Y. 2013. Affect analysis in context of characters in narratives. *Expert Systems with Applications*, 40/1, 168-176.
12. Soto, A., Cabrero, C., Menta, A. & Al, E. 2014. Estudio comparativo sobre el empleo de diccionarios en el análisis de sentimientos para textos cortos. *XVII Congreso Español sobre Tecnologías y Lógica Fuzzy (ESTYLF'14)*, Zaragoza, España, Feb.
13. Haddi, E., Liu, X. & Shi, Y. 2013. The role of text pre-processing in sentiment analysis. *Procedia Computer Science*, 17, 26-32.
14. Hassan-Montero, Y. 2010. Usabilidad de los tag-clouds: Estudio mediante eye-tracking. *SCIRE: Representación y Organización del Conocimiento*, 16/1, 15-33.
15. Strapparava, C. & Valitutti, A. 2004. WordNet-Affect: An affective extension of WordNet. In *4th International Conference on Language Resources and Evaluation (LREC'04)* (pp. 1083-1086).

ACKNOWLEDGEMENTS

This paper is the result of the work of the SOMOS (SOftware - MOdelling - Science) research group, which is funded by the Dirección de Investigación and Facultad de Ciencias Empresariales of the Universidad del Bío-Bío, Chile. This work was partially supported by Organic. Lingua, CIP-ICT-PSP.2010.6.2 - Multilingual online services, project reference: 270999.

CLAUDIA MARTÍNEZ
DEPTO. INGENIERÍA INFORMÁTICA,
UNIVERSIDAD CATÓLICA DE LA SANTÍSIMA CONCEPCIÓN,
UCSC ALONSO DE RIBERA 2850, CONCEPCIÓN, CHILE.
E-MAIL: <CMARTINEZ@UCSC.CL>

JORGE FERNÁNDEZ
DEPTO. INGENIERÍA INFORMÁTICA,
UNIVERSIDAD CATÓLICA DE LA SANTÍSIMA CONCEPCIÓN,
UCSC ALONSO DE RIBERA 2850, CONCEPCIÓN, CHILE.
E-MAIL: <JAFERNANDEZ@ING.UCSC.CL>

ALEJANDRA SEGURA
DEPTO. SISTEMAS DE INFORMACIÓN,
UNIVERSIDAD DEL BÍO-BÍO, UBB
AVDA. COLLAO 1202, CONCEPCIÓN, CHILE.
E-MAIL: <ASEGURA@UBIOBIO.CL>

CHRISTIAN VIDAL-CASTRO
DEPTO. SISTEMAS DE INFORMACIÓN,
UNIVERSIDAD DEL BÍO-BÍO, UBB
AVDA. COLLAO 1202, CONCEPCIÓN, CHILE.
E-MAIL: <CHVIDAL@UBIOBIO.CL>

CLEMENTE RUBIO-MANZANO
DEPTO. SISTEMAS DE INFORMACIÓN,
UNIVERSIDAD DEL BÍO-BÍO, UBB
AVDA. COLLAO 1202, CONCEPCIÓN, CHILE.
E-MAIL: <CLRUBIO@UBIOBIO.CL>

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 81–102
Received 17/02/2015, Accepted 14/05/2015, Final 18/06/2015.
ISSN 0976-0962, http://ijcla.bahripublications.com

Extracting Sentences Using Lexical Cohesion for Arabic Text Summarization

HAMZA ZIDOUM
AHMED AL-MAAMARI
NASSER AL-AMRI
AHMED AL-YAHYAI
SAID AL-RAMADHANI
Sultan Qaboos University, Muscat, Sultanate of Oman

ABSTRACT

Automatic Text Summarization has received a great deal of attention in the past couple of decades. It has gained a lot of interest especially with the proliferation of the Internet and the new technologies. Arabic as a language still lacks research in the field of Information Retrieval. In this paper, we explore lexical cohesion using lexical chains for an extractive summarization system for Arabic documents.

INTRODUCTION

Summary as defined by [21] is a "text that is produced from one or more texts, that conveys important information in the original text(s), and that is no longer than half of the original text(s) and usually significantly less than that." Summarization dates back to the late fifties where the first attempts relied entirely on statistical approaches. The sentence consisting of words with a high frequency were given a higher weight than the others indicating the importance of these sentences. Other than the mentioned approach, many different approaches were devised to tackle the problem of summarization [4], [16]. Cue phrases and lead method are one of the many approaches, the former extracts sentences containing words or phrases for e.g., "significant," "In

this paper" etc. The latter extracts first sentences of paragraphs assuming they contain the main idea. These methods rely on shallow approaches to indicate the importance of sentences to be included in the summary. Other approaches look at deeper levels like similarity that occurs when two words share a common stem, as in for instance the *thesaural* relationships that identify the different semantic relations existing between words, or the Rhetorical Structure Theory which identifies the relationship between text units.

There have been a few studies done on summarizing Arabic documents. Lakhas [6] attempted generating summary using a hybrid approach. The developed system relied on shallow approaches; frequency calculation, indicative expressions (cue words), lead method and title method. The system was evaluated in DUC 2004 (Document Understanding Conference). Systems that produce user focused summaries such as the one developed by [10] generates query based and concept based summaries. Arabic Query Based Text Summarization System (AQBTSS) is a query based single document summarizer that generates summaries relevant to the user query. Each sentence in the document is compared against the user query and only the relevant sentences are extracted. The other system Arabic Concept Based Text Summarization System (ACBTSS) is a concept based summarizer that generates a summary by matching each sentence against a set of keywords entered by the user, and these words represent a specific concept. The system uses a vector space model (VSM) that makes use of two measures; term frequency (tf) and inverse document frequency (idf) to weighing sentences.

A different approach by [12] was devised using clustering techniques. In this technique the roots are extracted for each word and are placed in the appropriate cluster. The words are assigned weights based on the number of words in the cluster it belongs. In addition to that it makes use of cue words which can enhance the weight of the sentence. The system then extracts sentences with highest scores, and the number of sentences depends on the size of the document.

Summarization can be described as a two-step process: (1) Building a source representation from the original text, and (2) Converting the source representation to an intermediate representation.

The input to the summarization systems can be in the form of textual data or other types of multimedia such as audio, video or images. Furthermore, summarization can even be performed on single documents or multiple documents consisting of a single language or more than one language also called multi-lingual. Output of the summarization system can be categorized into two groups: extracts and abstracts. Extract summaries consist of sentences from the original document whereas abstract summaries paraphrase some sections of the text or formed from the generated sentences. This requires language generation techniques and has some challenges.

Presenting a user with an adequate summary requires capturing the main theme of the document. This can be accomplished by looking at the related words in the document. Lexical cohesion can be created by semantically related words and represented by lexical chains. Lexical chains groups together semantically related words.

In comparison to English, and despite the recent interest due to geopolitic issues, Arabic still lacks research in the field of Information Retrieval. The factors contributing to this challenge of automatic processing of Arabic is the Arabic script itself due to the lack of dedicated letters to represent short vowels, changes in the form of the letter depending on its place in the word, and the absence of capitalization and minimal punctuation, Arabic words can be ambiguous as diacritics have been disappearing in contemporary writings [11]. Another factor is the normalization due to the inconsistency in the use of diacritic marks and certain letters in contemporary Arabic texts. Some Arabic letters share the same shape and are only differentiated by adding certain marks such as a dot, a hamza or a madda placed above or below the letter. For example, the “alif” in Arabic (ا) may be three different letters depending on whether it has a hamza above as in (أ) or a Hamza below as in (إ) or a madda above as in (آ). Recognizing these marks above or below a letter is essential to be

able to distinguish between apparently similar letters. But texts written in MSA often do not incorporate voweling as mentioned earlier nor do they adhere to the "proper" inclusion of marks above or beneath some Arabic letters. To manage this problem, the common practice in Arabic NLP systems is to normalize the input text [14]. For example, in order to handle the different variations in Arabic script, Larkeyand Connell (2001) replace the initial alif with a hamza above or below withsimply an alif, or bare alif. They also normalize the alif madda with a bare alif.Further, they normalize the final taa marbuuTa (ةor ىة) with a final haa (ەorىه) and the alif maqsuura (ى) with the yaa (ي).

In this paper we implement an algorithm that generates an extractive summary for Arabic single document using lexical chains. It makes use of WordNet [19], a lexical resource database containing nouns, verbs, adverbs and adjectives and groups each of them into set of synonyms called *synsets*. These synsets are related to others via semantic relations. The system will also make re-use of a parser and part-of-speech tagger to identify nouns.

The rest of the paper is organized as follows: Section 2 discusses previous works on lexical chains especially in the context of Summarization. Section 3 gives an overview of lexical cohesion and lexical chains. Section 4 presents the system architecture. Finaly section 5 is dedicated to discussing the results.

2. RELATED WORKS

Lexical chains was first introduced by [20]. They did not implement this algorithm as a machine-readable dictionary was not available then.

Barzilay & Elhedad [2] attempted to implement this algorithm using Wordnet. They performed segmentation on the source text and built lexical chains in every segment. All the interpretations were computed for every sense of the word to determine the correct one. This gave it an exponential runtime complexity even for short documents.

[22] implemented a linear time algorithm to compute lexical chains. The computation was done by creating a structure to store the interpretations without actually creating them. This two phases algorithm built an internal representation in the first phase and computed the chain score and performed Word Sense Disambiguation (WSD) in the second phase.

This algorithm was not yet accurate enough when performing word sense disambiguation as claimed by [8]. The algorithm divided the computation of lexical chains into three steps as opposed to two done by Silber and McCoy. In the first step, similar to Siber and McCoy they built a representation of all possible interpretations. This was the only time they made a pass through the document. The latter steps depended on this representation. The following step performed disambiguation where the word was assigned a correct sense based on sum of the weights leaving all the senses of the word, and this depends on the type of relation and the distance factor. The final step is the actual building of the lexical chains where the word whose sense is different than the assigned sense is removed. This algorithm was compared against Barzilay et al. [2] and Silber et al. [22] algorithm and performed better in terms of word sense disambiguation.

Medelyan [18] came up with a new approach to compute lexical chains with graph clustering. Lexical chains are treated as graphs where the nodes are the terms and the edges represent the relations. The chains cohesive strength is computed by the diameter of the graph; which can be defined as the longest of the shortest distance between any two nodes in the graph. This diameter is strong if it is fully connected where each node is connected to all the others, weak or moderately cohesive. The weak cohesive chains are decomposed into several highly cohesive chains by using a graph clustering algorithm. After strong chains are identified by using the chain scores, sentences are retrieved by summing up the weights of all the words in the sentence which correspond to the chain weights in which these words occur.

The previous methods assume that each chain represents a specific topic, but [7] claimed that a single chain cannot represent a whole topic. A cluster of lexical chains might represent a specific topic where they could define the "what," "where" and "why." They computed lexical chains using Galley et al. algorithm and filtered out the weak chains that had a score lower than a defined threshold. The remaining chains represent the strong ones and are clustered using cosine similarity. The connected sequences of sentences in these clusters are identified as segments and are scored. The best scoring segments are retrieved and the first sentence of each segment is included in the summary. The number of sentences to be retrieved in the summary depends on the number of unique sentences in the best scoring segments as only one sentence per segment is selected so this number could be less than the number of sentences required in the summary.

Cohesion introduced by Halliday and Hasan (1976) is a device used to *glue* together different parts of the text by means of cohesive relations, thus giving it a sense of unity. These relations can be classified into *co-reference*, *ellipsis*, *conjunction* and *semantic word relations*. Lexical cohesion can be created by semantically related words and is one of the most identifiable types of cohesion. Halliday and Hasan classified lexical cohesion into *reiteration* and *collocation*.

Reiteration can be achieved by *repetitions*, *superordinates*, *subordinates* and *synonyms*. *Collocation* defines semantic relations between words that often tend to occur in similar lexical contents (e.g., "The girl was digging the garden"). *Collocation* as a relation is more problematic to identify than reiteration.

Lexical cohesion does not occur between a pair of words but over a number of related words spanning a topical unit of text. This grouping of words is called lexical chains. To identify a relationship between words we make use of Arabic Wordnet.

Wordnet is a lexical resource database consisting of syntactic categories such as nouns, verb, adjective and adverbs. It groups words in each syntactic category into a set of synonyms called *synsets*. Furthermore, the synsets can be related to others in terms of semantic relations. Wordnet identifies the following relations:

synonyms are the basic and most important relation in Wordnet. Words are synonymous if they share the same sense and can be used interchangeably in the text. *Hypernyms/Hyponyms* also known as superordinate and subordinate. *Antonyms*(opposing name) are mainly described for adjectives and adverbs and *meronym/holonym* known as sub-part/whole part.

3. SYSTEM ARCHITECTURE AND IMPLEMENTATION

Automatic extractive summarization is a complex task that involves several challenging subtasks. The performance in each of these subtasks affects the performance for generating high quality summaries. First, Aramorph [3] proved to be faster than Stanford Parser [17]. Moreover, Aramorph works as Part of Speech tagger as well as word stemmer. The Aramorph module offers a better API which facilitates the integration. Diacritics add difficulties in comparing words and identifying their relations using Arabic Wordnet, so we used "Diacritic Remover" (reference) module to remove diacritics. Also we used "Arabic stem analyzer" to extract the stems. In general there is a lack of electronic lexical resources for Arabic, for example Arabic Wordnet is not as rich as the English counterpart for Wordnet, and the only available JAVA API for the Arabic Wordnet is is written by Abobakr Ahmed, available from sourceforge: <http://sourceforge.net/projects/javasourcecodeapiarabicwordnet/>.

The design of the system can be summarized in the following steps as illustrated in Figure 2 below. Each module takes as input the file produced by the previous module starting with the Tokenization module which takes a single Arabic text file.

1. Tokenization: Process where each sentence is partitioned into a list of tokens. Before the tokenization process we break the text into sentences. We adopted the Stanford document pre-processor module.
2. Part of Speech Tagging: It consists of classification of the tokens according to the best part of speech they represent; noun, verb, adverb, etc. Arabic stem analyzer which is called

Aramorph is used in this step. Aramorph PoS tagger produces the standard tag set as well as the extended tag set, so a noun could be assigned a tag set NN indicating a noun, ADJ indicating Adjective or NNS indicating a singular noun. Aramorph then returns the stem of the original word. The outputs are the stemmed words with diacritics.

3. Noun Filtering and Normalization: Nouns need to be filtered out prior to computing lexical chains. Regular Expressions have been used to accomplish this task by identifying the tags assigned as nouns by the toolkit then, we used "Arabic normalizer" to normalize stems as follows:

   - Normalization of *hamzated alif* to a bare alif.
   - Normalization of taa marbutah to haa.
   - Normalization of dotless yaa to yaa.
   - Removal of tatweel (stretching character).

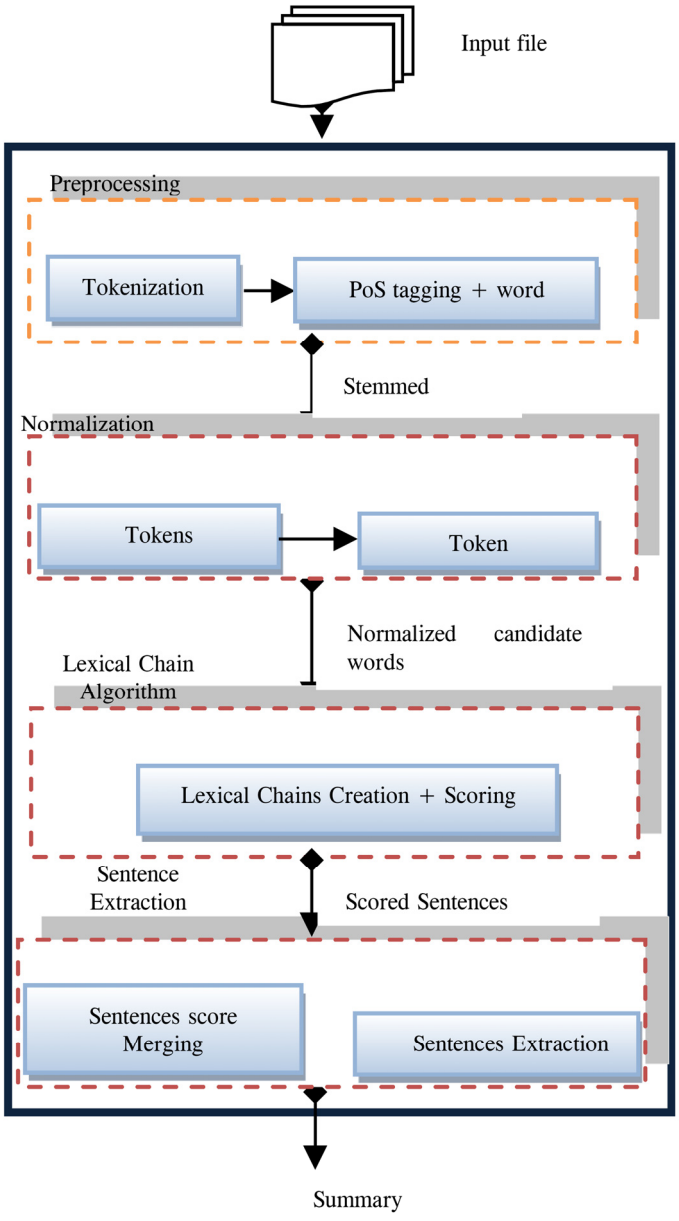


Figure 1. *System Architecture*

علمالحاسوبأوالمعلوماتية (الجزائر) أوالإعلامية (تونس) أوالمعلوميات (المغرب)
يدرسالحوسبةومعالجةالبياناتوالنظرياتوالتطبيقاتالتيتشكلالأساسلأتمتةنقلالمعلوماتوتشغيلهاوتحويلها،وذلكبدراسةبرمجياتالحاسوبوعتادالحاسوببشكلعلمي مجرد.
فيبعضالدولالعربيةيطلقعلىمصطلحعلمالحاسبالآليالمعلوماتيةاختصاراًوليس بقصدخلطهمعالعلومالأخرىوخاصةالتخصصاتالمتعلقةبتكنولوجياالمعلوماتالمهتمةبالتطبيقغيرالمبنيعلىأسسعلمية،كمايُطلقعليهفيالجزائراسم "الإعلامالآلي".
يبحثعلمالحاسوباستخداماتالحوسبةبجميعاشكالهالحلالمشكلاتمنمنظورعلميريا ضي.
وغالبًامايشملذلكتصميموبرمجةالبرمجياتلكيتستعملكأداةلحلهذهالمشاكل.
علمالحاسوبليسمعنيًابتعلمطريقةاستخداماالبرمجياتبشكلعاموبحدذاتها. منالصحيحالقولأنهناكبعضالوظائفالتيتعتمدبشكلأساسيعلىبعضالبرمجياتكبرمجياتالتصميملمصممينالجرافيكأومحرراتالنصوصوالجداوللمدخلياليبانات،لكنعلمالحاسوبليسمعنيًابدراسةطريقةالتعاملمعهذهالبرمجياتوغيرهابشكلعامو ليسمعنيًاكذلكبتصميمصفحاتالويبأوتجهيزها.
عندالحديثعنالبرمجياتفإنعلمالحاسوبيُعنى "بطريقة"بناءالبرمجياتبناءعلىأسسعلميةوريادةوبدراسةالخوارزمياتالأنسباستخدامًافيتلكالبرمجيات.
أصبحعلمالحاسوبعلمًاقائمًابذاته،يُعنىببحثأمورالحسابوالاحتسابمنمنظورعلميدقيق.
أماتكنولوجياالمعلوماتفهومجالآخريُعنىبمسائلأخرىمثلطُرُقاستخداماالبرمجياتوالتعاملمعهاوطرقاستعمالالمعلوماتأوحتىطريقةاستخدامماهوجاهزفيأغلبالأحيانلإنجازعملما،وغالبًامايُستخدممصطلحتكنولوجياالمعلوماتبشكلواسعبينالعامةوفيسوقالعمل.

Figure 2. *Arabic text input sample*

We used "*DiacriticRemover*" to remove Diacritic of the words (1) remove excessive whitespace sequences in the Arabic text, and (2) remove all diacritic marks like (TANWEEN_ALFATHA, TANWEEN_ALDAMMA, TANWEEN_ALKASRA, FATHA,

DAMMA, KASRA, SHADDA, SUKUN). The Output is the Normalized candidate words.

Word	Normalized Word
عِلْم	علم
حاسُوب	حاسوب
مُغَرَّب	مغرب
مُعالِج	معالج
بَيان	بيان
نَظَرِيّ	نظري
تَطْبِيق	تطبيق

3.1. *Lexical chain computation*

For the first step, we first process all the sentences to produce a chain list. For each candidate word in each sentence, we try to add the candidate word to a chain. If the candidate word hasalready been added to a chain, we increase the chain weight by adding *weight1*. This number represents a strong relation and a repetition of the word. Then we add the sentence id of the candidate word to the chain and update the score of the sentence inside the chain. In case the word is not added to a chain, we create a new chain. After creating the new chain, we generate a sense list of the word by using Arabic WordNet (AWN) and add this list to the newly created chain and we mark the word as a used word by adding it to a list named used words. Finally, if the chain sense list is not empty, we add the chain to our chain list. Otherwise, we ignore the chain, since it does not contain any sense list and it cannot create any relation with other chains. Examples of lexical chains are given in Figure 2.

490	[حاسوب, حاسب, قصد, حاسوب, مصطلح, خاص, حل, علي, مشكل, منظور, تعلم, تصميم, ...
570	[قصد, حاسوب, مصطلح, خاص, حل, علي, مشكل, منظور, تعلم, تصميم, طريق, برمج, حد, ...
310	[خاص, منظور, تعلم, علمي, علمي,]
120	[مصطلح, نظري, مشكل, بيان, قول, صفح, مجرد, مجرد,]
450	[مصطلح, حاسوب, جميع, خاص, علي, حل, مشكل, منظور, تعلم, تصميم, طريق, برمج, حد, ...
1870	[علم, حاسوب, مصطلح, حاسوب, خاص, علي, حل, مشكل, منظور, تعلم, تصميم, طريق, ...

Figure 3. *Lexical chains and their weights*

After constructing the chains, we try to find relations between them. For each chain, we create a new thread to create a relation list of the chain with the other chains. We have two kinds of relations with different weight, IN-relation and OUT-relation. To simplify these relations, let us say that we have two words: "word1" and "word2." If we could find a relation between the "word1" and one of the senses of "word2," than this will represent an IN-relation for "word1" and an OUT-relation for "word2." The ID of the chain that contains the "word1" will be added to the chain of the "word2" and vice versa. The IN-relation adds *weight2* to the weight and it is considered as a medium relation. The OUT-relation is considered as a weak relation and it adds *weight3* to the weight. Notice that *weight1*> *weight2*> *weight3*.

```
for each sentence { //First step
        for each word {
                if (word is repeated)   // already in UsedWord list
                {
                        add weight1 to chain weight; // weight1for repetition
                        add sentence id to chain;
                        update sentence score;
                }
                else {
                        Create new chain for the candidate word;
                        Generate a senseList taking all relations from Wordnet;
                        Add word to UsedWord list;
                        if (new chain senseList is not empty)
                                Add chain to chainList;
                }
        } // end words loop
} // end sentences loop
for each chain { // Second step
        Inside a new thread {
                if (chain word has a relation with other chain senseList) {
                Link chain with other chain;
                        add weight2 to chain weight; // IN relation
                        add weight2 to the other chain weight; // OUT relation
                } // end if
        }  // end thread
 }  // end chains loop
```

3.2. *Strong chains identification and extraction*

By this stage, each chain contains a chain number, a list of sentence numbers and weight. The algorithm for the sentences scoring and extracting is in Figure 6. Before extraction, we

calculate sentences score by adding a fraction of chain weight to each sentence in the chain. The fraction is to take an equal fixed part of the chains weight instead of taking the whole weight which sometimes can be large numbers.

After adding the final scores of sentences inside each chain, we reach the extraction stage. In this stage we have two main steps: (1) Merging the sentences from all chains with their scores to be extracted, and (2) Sentences with highest score will be extracted for the user depending on the extraction rate that he needs. The number of the sentences which will be extracted will be counted as follows:

```
Add 3% of each chain weight to score of sentences;
Create table for sentences and their scores;
For each chain {
        For each sentence {
                If (sentence in table)
                        Increment sentence score in the table;
                Else
                Add new record to the table;
        }
}
Sort table by the score;
Extract of sentences according to rating;
```

Number of extracted sentences =
*The ceil of (Extraction rate * Number of sentences in original text)*

For example, if the user needs 10% of extraction and the original text includes 22 sentences. The number of the sentences which will be extracted is:

Number of extracted sentences = 0.10 22 = ceil (2.2) =3 sentences.*
(Note: we used the ceil to prevent losing part of sentences)

The output is a combination of sentences with high score in the correct sequence which represent the final summary.

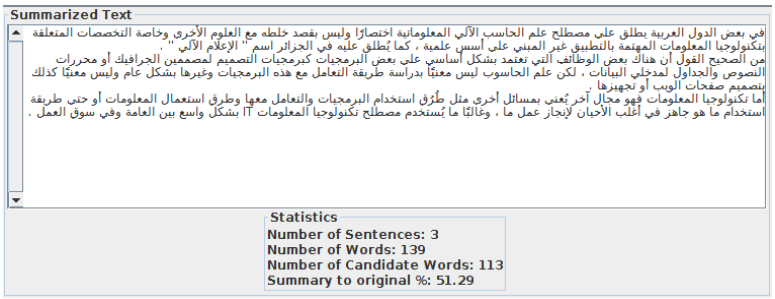


Figure 4. *Extracted Sentences*

The system graphical user interface is given in Figure 4, where the most prominent items are, for instance, loading a text file for summarization, original text, extracting the summary, e**xtraction rate in percentage,** displaying the generated summery, o**riginal text statistics (n**umber of sentences, number of words, number of candidate words), **summarized text statistics,** candidates words from the original text, n**ormalized words, their frequency,** senses from Arabic wordnet**, and the** weight of candidate words' lexical chains from the original text, lexical chains, candidate word's, sentences from original text and their weight**,** the score of sentence from the original text.

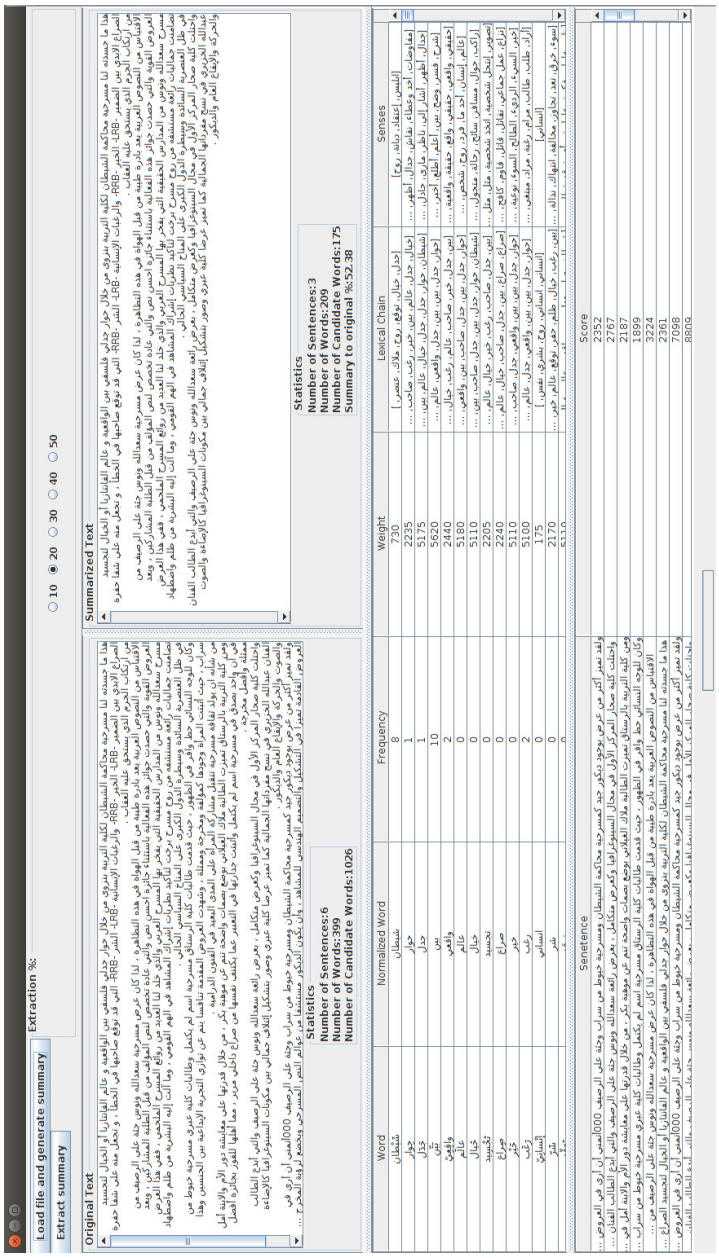


Figure 5. *Graphical User Interface (GUI)*

4. SUMMARY AND DISCUSSION

Evaluating summaries and automatic text summarization systems is not a straightforward process. When evaluating a summary, generally we measure two properties: the Compression Ratio and the Retension Ratio, i.e. how much of the central information is retained. We can also evaluate the qualitative properties of the summaries, such as how coherence, and readability. This is usually done by using a panel of human judges [13].

4.1. *Comparing with human summaries*

We measured how our system performed relative to different human summaries. We choose a sample text file which includes 34 sentences in total. First, we generate summaries using our system for this text in different extraction rates which is as follows:

Table 1. *Number of summary sentences*

Rate	#number of sentences un the summary
10%	4
20%	7
30%	10
40%	14
50%	17

We gave five native english-speaking persons the same sample to summarize. Let's call them person1 to person5.

Total # of sentences in the original text is = 34 sentences.
A: # of our project's summary sentences (*differentiate depending on extraction rate*).
p1: # of person1 summary sentences = 12
p2: # of person2 summary sentences = 17
p3: # of person3 summary sentences = 14
p4: # of person4 summary sentences = 18
p5: # of person5 summary sentences = 19

Comparing human summaries with our system summary, we have found that the intersection between the summary sentences of two of them is as shown in Table 2. (Note: The intersections represent the number of sentences which are common in human summary and our system's summary)

Table 2. *Human and our system summary intersection*

Ex. rate	A	A ∩ p1	A ∩ p2	A ∩ p3	A ∩ p4	A ∩ p5
10%	4	4	3	1	1	2
20%	7	6	6	4	5	4
30%	10	8	7	9	8	6
40%	14	10	9	12	11	13
50%	17	11	12	13	16	14

Finally, to measure the quality of our system depending on human summaries we have divide the intersection between the summary sentences of two of them by the total number of human summaries sentences in every percentage of extraction. The results are as shown in Table 3. We conclude that the accuracy of our system will increase when the rate of extraction is increasing.

Table 3. *Projects's summary quality comparing to human summaries*

Ex. rate	(A ∩ H1)/ H1	(A ∩ H2)/ H2	(A ∩ H3)/ H3	(A ∩ H4)/ H4	(A ∩ H5)/ H5
10%	0.333	0.176	0.071	0.056	0.105
20%	0.500	0.353	0.286	0.278	0.211
30%	0.667	0.412	0.643	0.444	0.316
40%	0.833	0.529	0.857	0.611	0.684
50%	0.910	0.705	0.929	0.889	0.737

Figure 6 gathers the whole information and results of comparing our system with 5 human summaries using different extraction rate. The vertical coordinate show the quality percentage and the horizontal coordinate represents the intersection of ours system's summaries and human's summaries to human summaries ration using different extraction rate summaries.

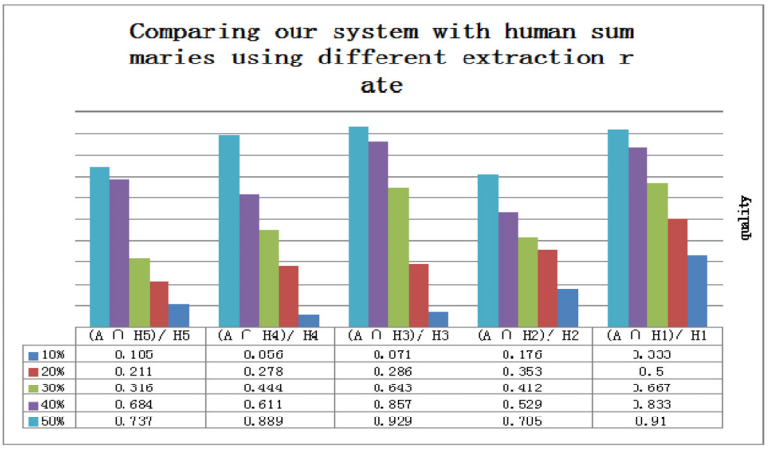


Figure 6. *Comparing ours system with human summaries using different extraction rate*

How the extraction rates effects the result? Having high extraction rates like 60% for example in small texts does not make sense because in such a case the summary may contains the most of those texts which is not reasonable. On the other hand, having small extraction rates like 20% in a summarization of large texts may result in loss of critical information of the original texts.

4.2. *Evaluation using human judgment*

Using human judgments to have another way of evaluating a summary even that the expensive cost of human judgment. We ask 7 different human to judge the summaries of our system using 6 different texts from different categories which are culture, economy, international, local, religion and sport.

First, we generate summaries for those 6 texts with different sizes and we get different numbers of summary sentences as follows:

# of summary sentences using culture text = 8
# of summary sentences using economy text = 10
# of summary sentences using international text = 7

# of summary sentences using local text = 5
# of summary sentences using religion text = 9
# of summary sentences using sport text = 4

Then, we give these all summaries for each human involved in this evaluation and we ask them to count the number sentences that describe the entire text to the total number of summary sentences in different types of texts. Finally, we find the ratio of these numbers of sentences to the total number of sentences in the summaries.

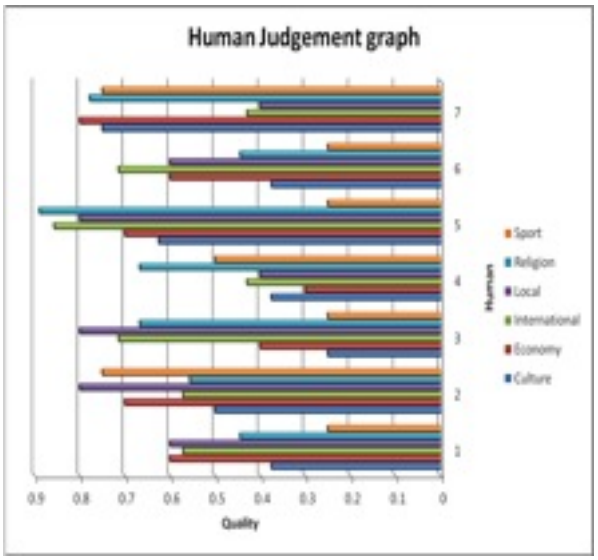


Figure 7. *Human judgment on six texts*

Table 4. Number of sentences that describe the entire text to the total number of summary sentences in different types of texts

	Culture	Economy	International	Local	Religion	Sport
Human1	0.375	0.600	0.571	0.600	0.444	0.250
Human2	0.500	0.700	0.571	0.800	0.556	0.750
Human3	0.250	0.400	0.714	0.800	0.667	0.250
Human4	0.375	0.300	0.429	0.400	0.667	0.500
Human5	0.625	0.700	0.857	0.800	0.889	0.250
Human6	0.375	0.600	0.714	0.600	0.444	0.250
Human7	0.750	0.800	0.429	0.400	0.778	0.750

Figure 7 shows human judgment of our system using 6 different texts from different categories which are culture, economy, international, local, religion and sport. We can see from the graph that the human judgment will depend on the type of the text.

REFERENCES

1. Abbas, M., Smaili, K. & Berkani. 2011. Evaluation of topic identification methods on Arabic corpora. *Journal of Digital Information Management*, 9/5, 185-192.
2. Barzilay, R. & Elhedad, M. 1997. *Using* lexical chains for text summarization. In proceedings of the *Intelligent Scalable Text Summarization Workshop (ISTS'97), ACL*
3. Buckwalter, T. 2001. ARAMORPH Arabic morphological analyzer. *Linguistic Data Consortium.*
4. Das, D. & Martins, A. 2007. A survey on automatic text summarization. *Literature Survey for the Language and Statistics Course at CMU.*
5. Doran, W., Stokes, N., Carthy, J. & Dunnion, J. 2004. Assessing the impact of lexical chain scoring methods on summarization. In proc of *CICLING'04* (pp. 627-635).
6. Douzidia, F. S. & Lapalme, G. 2004. *Lakhas, an Arabic Summarization System*.
7. Ercan, G. & Cicekli, I. 2008. Lexical cohesion based topic modeling for summarization. *CICLing 2008, LNCS 4919* (pp. 582-592).
8. Galley, M. & McKeown, K. 2003. Improving word sense disambiguation in lexical chaining. In proceeding of *18th International Joint Conference on Artificial Intelligence pages 1486-1488*
9. El-Haj, M., Kruschwitz, U. & Fox, C. 2011. Experimenting with automatic text summarization for Arabic. *Human Language Technology. Challenges for Computer Science and Linguistic* (pp. 490-499), Springer Berlin Heidelberg.
10. El-Haj, M., Kruschwitz, U. & Fox, C. 2010. Using mechanical turk to create a corpus of Arabic summaries. In the *7th International Language Resources and Evaluation Conference (LREC 2010)* (pp. 36-39), Valletta, Malta, LREC.
11. Farghaly, A. & Shaalan, A. 2009. Arabic natural language processing: Challenges and solutions. *ACM Transactions on Asian Language Information Processing*, 8/4, Article 14.

12. Haboush, A., Al-Zoubi, M., Momani, A. & Tarazi, M. 2012. Arabic text summarization model using clustering techniques. *World of Computer Science and Information Technology Journal*, 2/3, 62-67
13. Hassel, M. 2004. Evaluation of automatic text summarization. Licentiate Thesis. *KTH Numerisk analys och datalogi*, Stockholm, Sweden.
14. Larkey, L. & Connell, M. E. 2001. Arabic information retrieval at UMASS in TREC-10. In proceedings *of the 10th Text Retrieval Conference (TREC'01)*.
15. Lehal, G. 2010. A Survey of text summarization extractive techniques. *Journal of Emerging Technologies in Web Intelligence*, 2/3, 258-268
16. Jezek, K. & Steinberger, J. 2007. Automatic text summarization (The state of the art 2007 and new challenges). *Information Processing and Management*, 42/6.
17. Klein, D. & Manning, C. D. (2002). Fast exact inference with a factored model for natural language parsing. *In NIPS.*
18. Medelyan, O. 2007. Computing lexical chains with graph clustering. In proceedings of the *Association for Computational Linguistics* (pp. 85-90).
19. Miller, G. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38/11, 39-41
20. Morris, J. & Hirst, G. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Association for Computational Linguistics*, 17/1.
21. Radev, D., McKeown, K. & Hovy, E. 2002. Introduction to the Special Issue on Summarization. *Computational Linguistics*, 28/4, 399-408.
22. Silber, G. & McCoy, K. 2002. Efficiently computed lexical chains as an intermediate representation for automatic text summarization. *Association for Computational Linguistics*, 29/4, 487-496.

HAMZA ZIDOUM
DEPARTMENT OF COMPUTER SCIENCE,
COLLEGE OF SCIENCE,
SULTAN QABOOS UNIVERSITY,
MUSCAT, SULTANATE OF OMAN.

AHMED AL-MAAMARI
DEPARTMENT OF COMPUTER SCIENCE,
COLLEGE OF SCIENCE,
SULTAN QABOOS UNIVERSITY,
MUSCAT, SULTANATE OF OMAN.

NASSER AL-AMRI
DEPARTMENT OF COMPUTER SCIENCE,
COLLEGE OF SCIENCE,
SULTAN QABOOS UNIVERSITY,
MUSCAT, SULTANATE OF OMAN.

AHMED AL-YAHYAI
DEPARTMENT OF COMPUTER SCIENCE,
COLLEGE OF SCIENCE,
SULTAN QABOOS UNIVERSITY,
MUSCAT, SULTANATE OF OMAN.

SAID AL-RAMADHANI
DEPARTMENT OF COMPUTER SCIENCE,
COLLEGE OF SCIENCE,
SULTAN QABOOS UNIVERSITY,
MUSCAT, SULTANATE OF OMAN.

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 103–116
Received 19/02/2015, Accepted 18/05/2015, Final 14/06/2015.
ISSN 0976-0962, http://ijcla.bahripublications.com

Automated Evaluation of Short Summaries

DIEGO AGUIRRE [1]
ANTHONY MORSE [2]
OLAC FUENTES [1]

[1] *University of Texas, USA*
[2] *State University of New York, USA*

ABSTRACT

Children in elementary school are not only taught to read, but to understand what they are reading. To assess and improve their ability to understand concepts, students are often required to write short summaries of articles. Due to their nature, these documents often include misspelled words, missing punctuation, and erroneous grammatical structure. Evaluating these summaries is a laborious task that not only demands a significant amount of time from professors, but also limits the speed in which students can receive feedback. This paper presents a method for evaluating short summaries written by elementary school students. Our experiments show that incorporating semantic similarity/relatedness measures between words benefits the tasks of attribute selection and attribute weighting. We also show that preprocessing steps, such as the correction of misspelled words, are beneficial for the evaluation of short summaries. Our automatic grader has a mean absolute error of 0.98 when compared to a human grader on a 9-point grading scale. This agreement is comparable to the average agreement between two human graders.

Keywords: Natural language processing, machine learning, summary evaluation

1. INTRODUCTION

Students in elementary school are given many assignments to assess and improve their understanding of different pieces of text. In particular, young students are taught to identify the main idea of different articles along with the structure that authors use to convey ideas. For instance, students are often required to identify if the writer of an article is comparing two or more objects, or if the writer is presenting a problem and its solution. To evaluate their understanding, students are usually required to read an article on a computer and write a recall. In such recall, students are expected to state the main idea of the article along with supportive sentences that describe the structure that the author is using. Even though this process is done using a computer, professors are still required to read these summaries and grade them manually. This is a time-consuming task and makes it impossible to provide students with immediate feedback.

The automatic assessment of summaries has been studied by the text summarization community for several years. The objective is to evaluate summaries that are generated by automated tools. The methods employed usually compare fragments of the summary being evaluated against reference summaries produced by humans [4]. However, summaries written by elementary school students are different from those generated by automated tools. They are a couple of sentences long, have a significant number of misspelled words, and require a fast assessment to provide students with timely feedback. In this article, we present a method for evaluating this special type of summaries using text categorization and variable estimation techniques.

2. RELATED WORK

There has been much work in the field of automatic text summarization. A key task that researchers in this area have been studying for several years is the evaluation of automatically generated summaries. Lin and Hovy [11] addressed this problem

by introducing an evaluation mechanism for this type of summaries. Their idea is based on a scoring system used for the evaluation of machine translation systems called BLEU. This scoring mechanism measures how close automatically generated translations are to translations made by humans. To do so, they use the frequencies of n-grams that are common in both machine-generated translations and reference translations. Thus, BLEU is a measure of how well an automated translation overlaps with reference translations using co-occurrences of n-grams to make the comparison. Lin and Hovy propose to use this idea to evaluate machine-generated summaries. They found that using only unigrams instead of n-grams produced better results for their task. They claim that this is caused by the fact that n-grams tend to score for grammatical structure rather than content. Their results show that using co-occurrence statistics with unigrams produces assessments that are highly correlated with human assessments [11].

A limitation with the n-gram approach is that summaries with different content can be considered equally good. This is mainly due to the fact that people express similar ideas using different words. Harnly et al. [9] propose to use what is called the automated pyramid method to address this limitation. Their method addresses some characteristics associated with abstractive summaries. These are that summaries with the same quality not only have an overlap in content, but also have a unique contribution, and that wording to express the same content can unpredictably vary. The automated pyramid method requires having multiple reference summaries available. These summaries are used to identify text fragments, called contributors, that are believed to express the same meaning. These fragments are weighted based on their frequencies in the text objects. These contributors are used to create what is called a pyramid. When an unseen summary is evaluated, its text is compared with the pyramid of contributors to see if there are any candidate contributors in the unseen summary that express the same meaning as the contributors in the pyramid. These candidate contributors are weighted using their frequencies. Finally, the score of the summary is the ratio between the sum of

the weights of its candidate contributors and the sum of weights of an ideal summary. They define an ideal summary as a summary that uses the candidates from the pyramid with the highest weights and has the same size as the summary being graded. They compared their automated version of the pyramid method with its non-automated counterpart. They used Pearson and Spearman correlations as comparison metrics. Their results produced the values 0.942 and 0.943 for these correlation measures respectively [9]. This shows that the results given by this approach are very similar to the ones produced by human graders.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is another evaluation tool that uses human summaries to measure the quality of unseen summaries [10]. This mechanism is based on n-gram co-occurrence statistics and the extraction and weighting of what they call the longest common subsequence. ROUGE has also shown to provide grades that are highly correlated with human-assigned grades.

Even though the above-mentioned methods have shown to be accurate when compared to human graders, they are limited by the availability of reference summaries. Louis and Nenkova [12] propose an automated evaluation method that does not use these human models. They propose to evaluate summaries by directly measuring how close they are to the original text. Even though they introduce different mechanisms to perform this comparison, their results show that using Jensen Shannon divergence alone as a measure of similarity between the original text and the summary leads to a 0.9 correlation with human rankings for pyramid scores. This shows that automatic essay evaluation without the use of human models is at the very least promising.

3. EXPERIMENTAL DATA

A data set consisting of 7870 summaries written by elementary school students was provided by Penn State University. The article used for the summarization process had a length of 98 words. Each of the summaries was manually graded by a

specialist in a 9-point scale. The distribution of grades in the data set is presented in Table 1.

4. APPROACH

Our approach can be seen as a three-step process. We first preprocess the text and extract features and their values from the summaries and the original article. In the second step, we use a subset of the obtained features to create three binary classifiers that learn to separate summaries at different points in the grading scale. To train each of these classifiers, we used a threshold *t* to partition the set of summaries into two: those that had a grade smaller than the threshold, and the rest. We found that the following thresholds provided the best results: 2.5, 4.5, and 7.5. In addition to the binary classifiers, we train another classifier that computes the grade of summaries given the distribution of part-of-speech tags used in the text. As a last step, we use the outputs of these classifiers along with other text-complexity/high-level features to train final classifier. Figure 1 shows the configuration of the different classifiers and the extracted features.

Table 1. *Distribution of grades*

Grade	Num.	Instances Percentage
1	190	2.41%
2	3832	48.69%
3	111	1.41%
4	388	4.93%
5	99	1.26%
6	1838	23.35%
7	412	5.24%
8	648	8.23%
9	352	4.48%

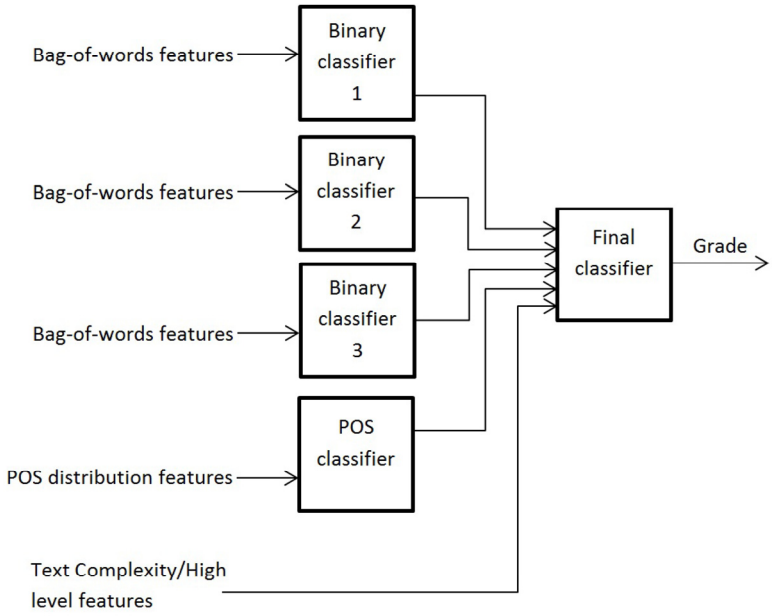


Figure 1. *Summary grading overall process*

4.1. *Data preprocessing*

Our dataset is composed of summaries that have a significant amount of misspelled words. We found that running a spell checker before extracting features from the text objects produced better results. We used the *Jazzy* library to Automated Evaluation of Short Summaries 5 programmatically replace all misspelled words in the summaries with correctly spelled ones. When a misspelled word was identified, a list of correctly spelled words was generated by the library. We replaced the misspelled word with the first suggestion, unless one of the other suggested words was a word used in the original article. In such case, we used the word found in the article to replace the misspelled word.

Additionally, we used Stanford's coreference resolution system to find all expressions that refer to the same entity in each summary and the original article. We replaced all mentions to an entity with the text that was first used when the entity was introduced in the text. For example, consider the following text

extract: *John loves to go mountain biking*. *He enjoys being outdoors*. The resulting text after out preprocessing stage would be the following: *John loves to go mountain biking. John enjoys being outdoors*.

4.2. *Binary classifiers*

We trained three classifiers that learned to segment the dataset into two. The first classifier learned to identify the summaries that had a grade greater than or equal to 2.5 from the rest. Similarly, the other two classifiers learned to partition the dataset dataset into two with 4.5 and 7.5 as the separating grades. We tested different learning algorithms to train these binary classifiers. We found that Multinomial Naive Bayes (MNB) and Support Vector Machines (SVM) were the ones that produced the best results.

For this binary classification task, we formed a bag of words to describe the elements in the dataset. We tried two approaches for selecting the words that would form the bag. In the first approach, we used all of the non-stop words in the original article that the students summarized. Notice that the bag was not formed using the set of words that the students used in their summaries. We found that using the words from the original article to form the bag produced better results. In the second approach, we extended the number of features by also incorporating all the bigrams that could be formed using all of the words in the original article. We used the TF-IDF measure to weight the attributes in the experiments where the SVM classifier was used as the binary discriminator. For the MNB classifier, we used the frequencies of the words as weights since the algorithm is designed to work with such frequencies.

We noticed that some summaries referred to the same concepts that the original article covered. However, the words that the students used to describe these concepts were not the same as the ones used by the author of the original article. As a result, we incorporated semantic and relatedness measures to influence how the frequencies of the words in the bag for a given summary are computed. The following pseudo-code describes

how the frequencies of the words in the bag are computed for a given summary *d*.

Algorithm 1: Formation of set T

```
for every word w_o in the bag of words
{
   freq_{w_o} := 0
   for every word w_d in summary d
   {
      if w_d = w_o OR sim(w_d, w_o) > 0.9
      {
         freq_{w_o} := freq_{w_o} + 1
      }
   }
}
```

To determine the similarity between two words, we used the adapted Lesk measure found in theWordNet: Similarity library. Lesk [1] proposed that the similarity of two words is proportional to the extent of the overlaps of their dictionary definitions. Banerjee and Pedersen [7] improved on this work by incorporating WordNet as the dictionary used for the word definitions. This similarity notion was improved once more by incorporating the network of relationships between concepts in WordNet. The implementation of this adapted Lesk measure is found in the WordNet::Similarity library.

4.3. *POS classifier*

We incorporated a classifier to estimate the grade of a summary given the distribution of the part-of-speech tags that it uses. To do this, we used Stanford's NLP library to extract all part-of-speech tags from summaries. We counted the frequencies of each possible tag for each summary. We normalized the information and used the distributions as vector representations of the summaries. The classifier that gave the best results for this task was a feed-forward neural network. The number of epochs used was 4000. The learning rate was set to 0.1 and the momentum was given a value of 0.5. The number of hidden layers was 3.

4.4. *Text-complexity/high-level features*
The following features were used in combination with the binary and POS classifiers to characterize each summary in the dataset.

- Number of words in the summary
- Number of sentences
- Number of misspelled words
- Average word length
- Euclidean distance between the original article and the summary using the bag of words weights as descriptors
- Percentage of words in the original article that appear in the summary
- Number of words longer than 5 characters
- Number of words longer than 6 characters
- Number of words longer than 7 characters

The output of the binary and POS classifiers along with the above-mentioned features were given as input to a final classifier. We tried two types of classifiers for this last step: A feedforward neural network and k-nearest neighbors. The number epochs used for the neural network was 4000. The learning rate was set 0.1 and the momentum was given a value of 0.5. The number of hidden layers was 3. For k-nearest neighbors, we found that k=3 produced the best results. The output of this final classifier was rounded since the grades assigned to the summaries are discrete.

5. EVALUATION

We are interested in comparing the grades given by our system to the ones assigned by the human grader. The following three metrics allow us to analyze this from different perspectives.

- Mean absolute error (MAE)
- Exact (E): number of summaries that were given the same grade as the human grader over the total number of summaries

– Adjacent (A): number of summaries that were given a grade that differed from the human grade by 1 point over the total number of summaries

To assess our approach, the data set was randomly split into two equal-sized subsets, preserving the distribution of the original grades. One of these two sets was used for training and the other for testing. Since our approach cannot not be directly compared to other approaches due to the uniqueness of the dataset, we developed a baseline approach where all extracted features were used for training. That is, a single classifier was trained using all features extracted from the training set and evaluated using the testing set. Comparing our approach to this simple baseline model allows us to recognize and approciate the utility of our summary evaluation system.

6. RESULTS

Table 2 shows the results obtained by our baseline. For each experiment, all attributes were used for training (BOW, POS, and text-complexity/high-level features). The column *Classifier* indicates the type of classifier used as the baseline. *BOW features* indicates how the bag-of-words was constructed. MAE indicates the mean absolute error of the experiment. *Exact* indicates the percentage of summaries that were given the same grade as the grade assigned by the human grader. *Adjacent* indicates the percentage of summaries that were given a grade that differed from the human grade by only 1 point.

Table 2. *Baseline results*

Classifier	BOW features	MAE	Exact	Adjacent
FFNN	Non-stop words	1.26	.44	.64
FFNN	Non-stop words + bigrams	1.28	.43	.66
SVM Regression	Non-stop words	1.17	.31	.69
SVM Regression	Non-stop words + bigrams	1.20	.30	.68
KN	Non-stop words	1.27	.38	.65
KNN	Non-stop words + bigrams	1.29	.35	.63

Table 3. *Proposed approach results*

Binary Classifier	BOW features	Final Classifier	MAE	Exact	Adjacent
MNB	Non-stop words	FFNN	0.98	.43	.77
MNB	Non-stop words + bigrams	FFNN	0.99	.42	.75
SVM	Non-stop words	FFNN	1.14	.35	.69
SVM	Non-stop words + bigrams	FFNN	1.16	.34	.68
MNB	Non-stop words	KNN	1.3	.35	.63
MNB	Non-stop words + bigrams	KNN	1.33	.35	.64
SVM	Non-stop words	KNN	1.4	.32	.65
SVM	Non-stop words + bigrams	KNN	1.41	.33	.66

Table 3 shows the results obtained when using binary and POS classifiers in combination with text-complexity/high-level features. Each row in the table represents an experiment. The column labeled *Binary Classifier* indicates the type of classifier that was used to partition the data set at the three different points in the grading scale. *BOW features* indicates what features were used to train the binary classifiers. *Final Classifier* indicates the type of classifier that was to ultimately estimate the grade of a summary. The columns *MAE*, *Exact*, and *Adjacent* have the same meaning as for Table 2.

7. DISCUSSION

Different conclusions can be drawn from the results. We observe that the feed forward neural network outperforms k-nearest neighbors in all instances. This can be easily attributed to the fact that neural networks, although not always, tend to outperform algorithms such as k-nearest neighbors in many estimation problems. It is also interesting to notice that the incorporation of bigrams did not have a significant effect in the obtained results. We expected the incorporation of bigrams to have a positive effect on all of the described metrics since other natural language tasks have been benefited from such process. We attribute this to

the nature of the problem we are solving. In other tasks, such as sentiment analysis, words such as “not” and “no” play a very important role. Thus, bigrams where these words appear tend to be appropriate attributes for the text objects. In our problem, students are meant to identify and write the main concepts that an article presents. Thus, using single words to form the bag suffices for the problem at hand.

We also observe that MNB is a suitable classifier for partitioning the dataset into 2 at different points in the grading scale. Although SVMs are suitable for binary classification problems, MNB showed to be a better candidate for this task. This same result has been observed in other problems where the classification problem involves text objects.

The results also show that our approach outperforms the baseline in terms of MAE using the best configuration that we found. More concretely, our best result was 0.98 in contrast to 1.17 from the baseline (a significant difference of almost 0.2). For the Adjacent metric, we also observe a remarkable difference between the two approaches, showing the robustness of the proposed approach. However, the baseline appeared to perform just as well as our approach for the Exact metric. In conclusion, we see that the combination of binary classifiers, a POS classifier, and other text-complexity/high-level features to train a final classifier outperforms the traditional approach of using all features to train a single classfier.

Finally, the results show that grading short summaries is a task that can be performed by a computer system reliably. In our best result, we see that our solution gave a grade that differed by at most one point from the actual grade for 77% of the cases. A one-point difference is something that is expected even when comparing two human graders. Although the grading of this type of summaries might still require human intervention to provide students with more concrete and detailed feedback, our solution can be of great use to quickly assess the quality of summaries written by young students. This can aid students when typing their summaries on a computer. Our solution can quickly analyze

the text and provide students with early feedback that they can use to improve their summaries before submitting them to the professor.

REFERENCES

1. Banerjee, S. & Pedersen, T. 2002. An adapted lesk algorithm for word sense disambiguation using wordnet. In *Computational Linguistics and Intelligent Text Processing* (pp. 136-145). Springer.
2. Cavnar, W. B. & Trenkle, John M. et al. 1994. N-gram-based text categorization. *Ann Arbor MI*, 48113(2), 161-175.
3. Feldman, R. & Sanger, J. 2007. The text mining handbook: advanced approaches in analyzing unstructured data. Cambridge University Press.
4. Hovy, E., Lin, C-Y., Zhou, L. & Fukumoto, J. 2006. Automated summarization evaluation with basic elements. In proceedings of *the Fifth Conference on Language Resources and Evaluation* (LREC 2006), (pp. 604-611). Citeseer.
5. Lewis, D. D. 1992. Feature selection and feature extraction for text categorization. In proceedings of the workshop on Speech and Natural Language (pp. 212-217). Association for Computational Linguistics.
6. Manning, C. D. 2011. Part-of-speech tagging from 97% to 100%: is it time for some linguistics? In *Computational Linguistics and Intelligent Text Processing* (pp. 171-189). Springer.
7. Pedersen, T., Patwardhan, S. & Michelizzi, J. 2004. Wordnet: Similarity: Measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004* (pp. 38-41). Association for Computational Linguistics.
8. Yang, Y. & Pedersen, J. O. 1997. A comparative study on feature selection in text categorization. In ICML, 97, 412-420.
9. Harnly, A., Nenkova, A., Passonneau, R. & Rambow, O. 2005. Automation of summary evaluation by the pyramid method. In *Recent Advances in Natural Language Processing (RANLP)*, (pp. 226-232).
10. Lin, C-Y. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop* (pp. 74-81).
11. Lin, C-Y. & Hovy, E. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In proceedings of the *2003 Conference of the North American Chapter of the Association for*

Computational Linguistics on Human Language Technology (pp. 71-78), Vol. 1, Association for Computational Linguistics.
12. Louis, A. & Nenkova, A. 2008. Automatic summary evaluation without human models. In *Notebook Papers and Results, Text Analysis Conference (TAC-2008),* Gaithersburg, Maryland (USA).

DIEGO AGUIRRE
UNIVERSITY OF TEXAS AT EL PASO,
EL PASO, TX 79902, USA.
E-MAIL: <DAGUIRRE6@MINERS.UTEP.EDU>

ANTHONY MORSE
STATE UNIVERSITY OF NEW YORK AT BROCKPORT,
BROCKPORT, NY 14420, USA.
E-MAIL: <ANTHONYMORSE92@GMAIL.COM>

OLAC FUENTES
UNIVERSITY OF TEXAS AT EL PASO,
EL PASO, TX 79902, USA.
E-MAIL: <OFUENTES@UTEP.EDU>

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 117–129
Received 24/02/2015, Accepted 19/05/2015, Final 17/06/2015.
ISSN 0976-0962, http://ijcla.bahripublications.com

Speaker Adaptation Applied to Sinhala Speech Recognition

THILINI NADUNGODAGE
RUVAN WEERASINGHE
University of Colombo School of Computing, Sri Lanka

MAHESAN NIRANJAN
University of Southampton, Highfield, UK

ABSTRACT

Sinhala, which the main spoken language of the majority of Sri Lanka, is an under-resourced language. Sinhala language is new to the speech recognition research field and faces the problem of not having suitable speech corpora available. For a language like Sinhala, it is essential to find out ways of developing good recognition models using a fewer sample of data. Speaker Adaptive methods provides the opportunity of improving speaker independent recognition systems into more speaker dependent systems by adapting the features of the user. In this paper we are presenting an experiment we carried out by adapting a pre-trained Sinhala speech recognition system (with a single voice) with several different speaker voices. Our experiment shows that although individual adaptation systems gives the best results for corresponding speakers, we can build general speaker adaptation models to get better results than building speaker independent models using the same amount of data.

1. INTRODUCTION

Sinhala, which is one of the national languages in Sri Lanka, is an under resourced language in the field of Automatic Speech Recognition (ASR) research. We have recently started looking

into Sinhala speech recognition and in the need of collecting data from the scratch since there are no previously created speech corpora are available. Collecting acoustic data is not a very difficult task and can be done by placing a recorder where people are talking. However, the hard part is that getting the corresponding text transcriptions. This is a very time consuming and a very tedious task so sometimes it is not very practical to transcribe a whole set of recorded data.

For several years Speaker Adaptation has been used to match the differences between the trained model features and the input data features. Speaker Adaptation techniques are mostly used to convert speaker independent ASR systems in to more speaker dependent ones. For adaptation one does not want large number of data samples as training a ASR model. Few utterances from a user is enough to get the system to respond to that user's voice satisfactorily.

Since speaker adaptation requires a fewer sample of new data, we thought of applying speaker adaptation for Sinhala language speech recognition as it is very hard to collect a large corpus of Sinhala speech data from scratch. In this paper we present how we used speaker adaptation in Sinhala speech recognition using a small set of data available.

Rest of the paper is as follows: Section 2 gives a brief description about Sinhala language. Section 3 reviews about speaker adaptation and different techniques that are used for it. In section 4 and 5 we presents our experiments, evaluation and results. Section 6 compares the speaker adaptation model with speaker independent model. Finally we conclude our paper in section 7 and future works in section 8.

2. SINHALA LANGUAGE

Sinhala is one of the official languages of Sri Lanka and the mother tongue of the majority (about 74%) of its population. Sinhala belongs to the Indu-Aryan language family. Sinhala language words can be divided into three main categories as *Nishapanna* (words that are of local origin), *Thadbhava* (Words borrowed from other languages in their near original form) and

Thathsama (Words derived from other languages but modified to be incorporated to Sinhala – mainly from Sanskrit and Pali). There is a high impact from Sanskrit in Sinhala given the fact that they are in the same language family. Pali has also a significant impact on Sinhala vocabulary. Tamil, Portuguese, Dutch and English have also impacted the structure and vocabulary of Sinhala due to various cultural, historical factors [1].

Spoken Sinhala contains 40 segmental phonemes; 14 vowels and 26 consonants. There are two nasalized vowels occurring in two or three words in Sinhala. Spoken Sinhala also has following several Diphthongs. Sinhala characters are written left to right in horizontal lines. Words are delimited by a space in general. Vowels have corresponding full character forms when they appear in an absolute initial position of a word. In other positions, they appear as strokes and, are used with consonants to denote vowel modifiers [2].

3. SPEAKER ADAPTATION

Speaker Adaptive model is an approach to obtain results which are nearly same as speaker dependent models without requiring a large amount of speaker specific data. In this process a trained model is tuned for a new speaker with relatively a few speech samples extracted from respective speaker. Speaker adaptation models have shown considerable improvement recognition over speaker independent models [3]. Speaker adaptation has become the modern interest in speech recognition because of its low cost approach. Speaker adaptation can be supervised or unsupervised, static or dynamic. In supervised adaptation, speech transcriptions are available and in unsupervised, it is not. In static adaptation, adaptation data is available prior to adaptation but in dynamic adaptation, data is incrementally available [4].

There are several statistical methods that are used for speaker adaptation. Some of them are described here.

- *Speaker normalization*
  Normalize the acoustic data to reduce mismatch with the acoustic models. One approach is to normalize the vocal tract length. Human vocal tract length varies according to their age, size, gender, etc. Frequency of human speech is inversely proportional to vocal tract length.
- *Maximum a posteriori (MAP) adaptation*
  Use the SI models as a prior probability distribution over model parameters when estimating using speaker-specific data. In this adaptation method it is required to have a well trained model to be adapted with new data. Also this requires a large amount of adaptation data since, it deals with separate phonemes.
- *Maximum likelihood linear regression (MLLR) adaptation*
  MLLR uses linear transformation of Gaussian model parameters to adapt to a given speaker. MLLR adaptation updates the mean vectors and covariance matrices using the new adaptation data.

There are several other methods such as Mixture Models which combines MAP and MLLR adaptation methods, Cluster Adaptive training, Eigenvoices, etc [5].

3.1. *Related work*

In literature there are lots of examples in applying speaker adaptation methods from better speech recognition. [6] and [3] presents MLLR speaker adaptation techniques and experiments. [7] presents a study of speaker adaptation techniques applied to hybrid HMM-ANN systems.

A acoustic-phonetic based speaker adaptation method based on decomposition of spectral variation sources is described in [8]. [9] presents a method for unsupervised instantaneous speaker adaptation by modeling the speaker variation in a continuous speech recognition system. A speaker adaptation system for limited data based on regression-trees is described in [10]. [11] and [12] presents how to use speaker adaptation methods for accent adaptation.

4. EXPERIMENT

Our experiment was carried out using a pre-built Sinhala speech recognition model which is considered as a base-line ASR model for Sinhala language. This base-line ASR model was trained using utterances from a single female speaker. The training data set was consisted with 3000 utterances (31,625 words), which were read speech of sentences extracted from the UCSC Sinhala Text Corpus [13]. The lexicon consisted with 4K unique words. The model was trained using the Hidden Markov Model Tool Kit (HTK) developed by the Cambridge University, UK [14]. The process of building this model is described in [15].

For speaker adaptation, we collected recorded speech from 5 female voices and 4 male voices. Each speaker read out 25 utterances. From these we used 10 utterances for adaptation and 15 remaining utterances for evaluation.

We have used HTK Toolkit's speaker adaptation tool and Maximum likelihood linear regression (MLLR) adaptation technique for these experiments.

We carried out the speaker adaptation experiment in 3 different ways to see how it performs with different sets of adaptation data. Following are the three methods we tried:

- Speaker Adaptation for individual speakers
- Speaker Adaptation for female voices / Speaker Adaptation for male voices
- Speaker Adaptation for both female and male voices

4.1. *Speaker adaptation for individual speakers*

In this experiment we did speaker adaptation in the general way which is to adapt the initial model with utterances from each speaker separately and built speaker dependent recognition models for each speaker.

4.2. *Speaker adaptation for female voices / speaker adaptation for male voices*

In this experiment we built two adaptation models separately for male voices and female voices. For the female adaptation model

we used adaptation data from only three speakers and for the male adaptation model we used adaptation data from only two speakers. Hence, in this experiment we have been able to evaluate the built models with previously unseen voices.

4.3. *Speaker adaptation for both female and male voices*
For this experiment we built only one adaptation model using both female (from 3 speakers) and male (from 2 speakers) adaptation data. As in previous experiment we were able to evaluate this model with both seen and unseen voices.

5. EVALUATION AND RESULTS

For these experiments we have not included out of vocabulary words in our adaptation data or evaluation data. Hence, all the words in the test data set were previously seen words.

Before doing the evaluation of adaptation experiments, we have evaluated the initial model (trained using single female speaker voice) using voices from different female and male speakers. Table 1 shows the word level accuracy values we obtained from this.

Table 1. *Evaluation of the Initial model with different male and female speakers*

Speaker	Word Accuracy
Female 1	36.69%
Female 2	26.43%
Female 3	3.60%
Female 4	20.14%
Female 5	0.80%
Male 1	0.00%
Male 2	0.00%
Male 3	1.52%
Male 4	0.00%

We can see that although the initial model can recognize other female voices (different from the trained voice) to some extent, it has failed in recognizing male voices.

5.1. *Speaker adaptation for individual speakers: Evaluation*
For this experiment we have used 15 utterances from each speaker as test data. Table 2 shows the recognition accuracy with respect to each voice (Each adaptation model we built were evaluated using the corresponding voice). We have compared the word level accuracy values with adaptation and without adaptation.

Table 2. *Evaluation of the individual adaptation models with corresponding male and female speakers*

Speaker	Word Accuracy	
	Without Adaptation	**With Adaptation**
Female 1	36.69%	75.54%
Female 2	26.43%	58.57%
Female 3	3.60%	29.50%
Female 4	20.14%	72.66%
Female 5	0.80%	72.66%
Male 1	0.00%	66.91%
Male 2	0.00%	39.20%
Male 3	1.52%	46.04%
Male 4	0.00%	33.81%

Table 2 clearly shows how the accuracy of recognizing each speaker's voice is increased although the adaptation was done using a very small sets of data as 10 utterances. We can see that most of the female voices perform with high accuracy with adaptation. Even the male voices (which are very different from the initial model's training voice) shows a significant accuracy increasing after the adaptation.

5.2. *Speaker adaptation for female voices / speaker adaptation for male voices: Evaluation*
Here we have evaluated the two separate models (Female adaptation model and Male adaptation model) with two sets of data. One with voices from the speakers we have used to adapt the models (previously seen data) and one with voices from new speakers (previously unseen data).

Table 3. *Evaluation of the male and female adaptation models with seen and unseen male/female speakers*

Test Set	**Word Accuracy**	
	Without Adaptation	**With Adaptation**
Female voices (seen)	22.3%	40.53%
Female voices (unseen)	10.98%	23.02%
Male voices (seen)	0.00%	54.37%
Male voices (unseen)	0.81%	12.23%

In Table 3 we can see that by building separate common models for male and female voices also we can obtain an increased accuracy. Although this performance may not be good as the performance of individual adaptation models, we can see that by this method even for new (unseen) speakers we can get a good recognition accuracy compared to the initial model.

5.3. *Speaker adaptation for both female and male voices: Evaluation*

To evaluate this general adaptation model we have used the same test sets we used in the previous experiment. In Table 4 we can see that even by building a general adaptation model for both male and female voices, we can obtain an increase in the accuracy than the initial model.

Table 4. *Evaluation of the general adaptation model with seen and unseen male/female speakers*

Test Set	**Word Accuracy**	
	Without Adaptation	**With Adaptation**
Female voices (seen)	22.3%	26.14%
Female voices (unseen)	10.98%	15.11%
Male voices (seen)	0.00%	19.01%
Male voices (unseen)	0.81%	7.55%

6. SPEAKER ADAPTATION VS. TRAINING SPEAKER INDEPENDENT MODELS

In the previous sections we have described the results we got from various types of speaker adaptation models. In this section we thought of comparing these results with results we can obtain by training a speaker independent model using the adaptation data we used for speaker adaptation.

Our initial training set was consisted with 3000 utterances from one female speaker. To this initial training set we added the data we used for speaker adaptation (90 utterances from 5 females and 4 males: 10 utterances each) to create a new training data set. Using this new train data we trained a new acoustic model which is no longer dependent on one speaker. We evaluated this model using the same test sets we have used for speaker adaptation evaluation.

Table 5. *Comparison of the performance of general adaptation model and speaker independent model*

Test Set	**Word Accuracy**	
	Speaker Independent Model	**General SA Model**
Female voices (seen)	16.07%	26.14%
Female voices (unseen)	7.91%	15.11%
Male voices (seen)	0.00%	19.01%
Male voices (unseen)	0.00%	7.55%

Table 5 shows a comparison of evaluation accuracy for general speaker adaptation model and speaker independent model. I clearly shows that speaker adaptation is far more better than building speaker independent systems where only a small sample of data is available.

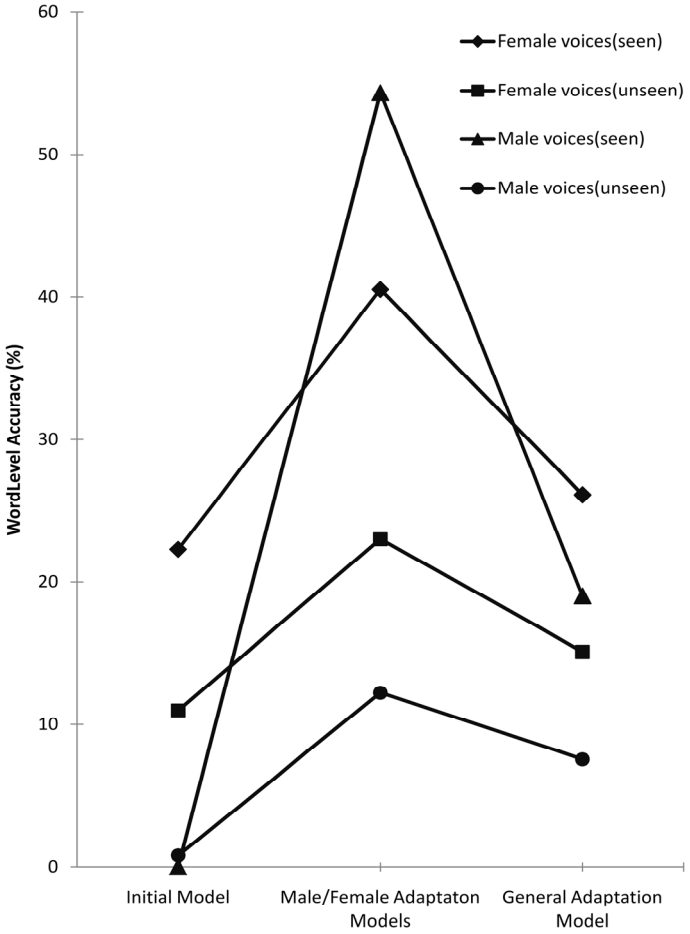


Figure 1. *Word Level performances of the Initial model, Male/Female Adaptations models and the General Adaptation model with different evaluation data sets.*

7. CONCLUSIONS

From the above experiments we can say that any level of adaptation can lead to increasing of recognition accuracy than the initial model. Figure 1 shows the summary of what we have

gathered in these experiments. We can say that although individual male / female adaptation systems are best to obtain a good recognition accuracy, we can built a general adaptation system where we are able to use for new speakers without prior adaptation. Also we have shown that by building a general speaker adaptation system we can achieve better recognition accuracy than building a speaker independent recognition model where there are fewer samples of data available. We can use speaker adaptation is suitable for an under-resourced language like Sinhala, where full corpora of transcribed speech is hard to come by.

8. FUTURE WORKS

The experiments and results presented in this paper were based on a very small data set with a very few vocabulary where we did not consider out of vocabulary words. As future work we intend to improve this work by collecting more data from more speakers and by increasing the vocabulary size and also considering out of vocabulary words in the evaluation sets.

REFERENCES

1. Weerasinghe, R., Wasala, A., Gamage, K. 2005. A rule based syllabification algorithm for sinhala. In *Natural Language Processing - IJCNLP 2005* (pp. 438-449), Springer.
2. Wasala, A., Weerasinghe, R., Gamage, K. 2006. Sinhala grapheme-to-phoneme conversion and rules for schwa epenthesis. In proceedings of the *COLING/ACL on Main Conference Poster Sessions* (pp. 890-897), Association for Computational Linguistics.
3. Ganitkevitch, J. 2005. Speaker adaptation using maximum likelihood linear regression. In Rheinish-Westesche Technische Hochschule Aachen, the course of Automatic Speech Recognition, www-i6. informatik. rwthaachen. <de/web/Teaching/Seminars/ SS05/ASR/Juri Ganitkevitch Ausarbeitung.pdf>.
4. Renals, S. 2013. Speaker adaptation (automatic speech recognition asr lecture 10 &11.
5. Shinoda, K. 2011. Speaker adaptation techniques for automatic speech recognition. *Proc. APSIPA ASC 2011 Xi'an*.

6. Leggetter, C. J. & Woodland, P .C. 1995. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. *Computer Speech & Language*, 9, 171-185.
7. Neto, J., Almeida, L., Hochberg, M., Martins, C., Nunes, L., Renals, S. & Robinson, T. 1995. Speaker-adaptation for hybrid hmm-ann continuous speech recognition system. In *Fourth European Conference on Speech Communication and Technology, EUROSPEECH* (pp. 18-21), International Speech Communication Association.
8. Zhao, Y. 1994. An acoustic-phonetic-based speaker adaptation technique for improving speaker-independent continuous speech recognition. *Speech and Audio Processing*, 2, 380-394.
9. Strom, N. 1996. Speaker adaptation by modeling the speaker variation in a continuous speech recognition system. In proceedings of *ICSLP 96* (pp. 989-992).
10. Wang, S., Cui, X. & Alwan, A. 2007. Speaker adaptation with limited data using regression-tree-based spectral peak alignment. *Audio, Speech, and Language Processing, IEEE Transactions on 15* (pp. 2454-2464).
11. Zheng, Y., Sproat, R., Gu, L., Shafran, I., Zhou, H., Su, Y., Jurafsky, D., Starr, R. & Yoon, S. Y. 2005. Accent detection and speech recognition for shanghai-accented mandarin. In *Interspeech Citeseer* (pp. 217-220).
12. Clarke, C. M .& Garrett, M .F. 2004. Rapid adaptation to foreign-accented english. *The Journal of the Acoustical Society of America*, 116, 3647-3658.
13. Weerasinghe, R., Herath, D., Welgama, V., Medagoda, N., Wasala, A., Jayalatharachchi, E. 2007. Ucsc sinhala corpus - pan localization project-phase i.
14. Young, S. 1993. The htk hidden markov model toolkit: Design and philosophy. Technical report, Department of Engineering, Cambridge University, UK.
15. Nadungodage, T. & Weerasinghe, R. 2011. Continuous sinhala speech recognizer. In *Conference on Human Language Technology for Development* (pp. 141-147), Alexandria, Egypt.

ACKNOWLEDGMENTS

Authors of this paper would like to acknowledge the National Research Council (NRC) of Sri Lanka for funding this research. They are also indebted to the research team members of the

Language Technology Research Laboratory of the University of Colombo School of Computing, Sri Lanka, for assisting in numerous ways.

THILINI NADUNGODAGE
LANGUAGE TECHNOLOGY RESEARCH LABORATORY,
UNIVERSITY OF COLOMBO SCHOOL OF COMPUTING,
SRI LANKA.
E-MAIL: <FHND@UCSC.LK>

RUVAN WEERASINGHE
LANGUAGE TECHNOLOGY RESEARCH LABORATORY,
UNIVERSITY OF COLOMBO SCHOOL OF COMPUTING, SRI LANKA
E-MAIL: <ARWG@UCSC.LK>

MAHESAN NIRANJAN
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE,
UNIVERSITY OF SOUTHAMPTON, HIGHFIELD,
SOUTHAMPTON SO17 1BJ, UK.
E-MAIL: <M.NIRANJAN@SOUTHAMPTON.AC.UK>

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 131–148
Received 27/02/2015, Accepted 23/05/2015, Final 18/06/2015.
ISSN 0976-0962, http://ijcla.bahripublications.com

Exploratory Study of Word Sense Disambiguation Methods for Verbs in Brazilian Portuguese

MARCO ANTONIO SOBREVILLA CABEZUDO
THIAGO ALEXANDRE SALGUEIRO PARDO
Universidade de São Paulo, São Paulo, Brasil

ABSTRACT

Word Sense Disambiguation (WSD) aims at identifying the correct sense of a word in a given context. WSD is an important task for other applications as Machine Translation or Information Retrieval. For English, WSD has been widely studied, obtaining different performances. Analyzing by morphosyntactic class, Verb is the hardest class to be disambiguated. Verbs are an important class and help to the sentence construction. Studies show that the disambiguation of verbs brings improvements into other applications. For Portuguese, there are few studies about WSD and, recently, these have been focused on general purpose. In the present paper, we report an exploratory study of knowledge-based Word Sense Disambiguation methods for verbs in Brazilian Portuguese, using WordNet-Pr (for English) as sense repository; and a comparison with the results obtained for nouns. The results show that, both All-words and Lexical sample evaluation, no methods outperformed the baseline. However, the multi-document scenario helped the WSD task.

Keywords: Word sense disambiguation, knowledge-based methods, Brazilian Portuguese

1. INTRODUCTION

Semantics is a deep linguistic knowledge level [1] and it is a popular subject in the Natural Language Processing community.

One of the most important problems related to Semantics is the ambiguity and, specifically, Lexical Ambiguity. Lexical Ambiguity occurs when a word may express two or more senses in a determined context. Lexical Ambiguity may be expressed in various difficulty levels. For example, consider the following four sentences:

- *homemcontou o número de pessoasqueficaramferidas* ("The man counted the number of people who were injured.").
- *O jogador bateu na bola com força* ("The player kicked the ball strongly.").
- *lutadorbateu as botas* ("The fighter died." or "The fighter kickedthe bucket.").
- *banco quebrounasemanapassada* ("The seat broke last week."or "the bank failed last week.").

In the first example, the sense of the verb "*contar*" may beeasily identified (to determine the total number of a collection of items); in the second example, the sense of the verb "*bater*" is easily identified too (to kick something); but, in the third example, the sense of the same verb may be difficult to identify, it could mean"to die" if we consider the expression "*bater as botas*", or could mean "to kick something" if we consider only the verb "*bater*"; finally, in the last example, it is necessary that we have more context and world knowledge, therefore, it is hard to identify the sense of the verb "*quebrar*", that could mean "to break an artifact (like seat)" or "to failfinancially (financial institution)".

Word Sense Disambiguation (WSD) aims atidentifying the correct sense of a word within a given context using a pre-specified sense repository [2]. WSD is considered an important task of other applications, as Information Retrieval, Machine Translation and Sentiment Analysis.

For English, there are many studies about WSD using different approaches and techniques [3]. Recently, knowledge-based WSD methods have become very popular [4]. This is due to the increase of the need of General Purpose WSD methods,

which are able to be integrated into any application and domain. Despite the increasing of this popularity, studies have shown that WSD is a hard task to be solved and the obtained performance is not high. Analyzing by morphosyntactic class, verb is the hardest class to be disambiguated. The main reason for this difficult in verbs is that WSD methods use, generally, surface information to disambiguate a word, but verbs need syntactic and semantic information to get better results.

According to [5], verbs are an important class and help to the sentence construction. Studies show that the disambiguation of verbs brings improvements into other applications as Semantic Role Labeling [6].

For Portuguese, there are few studies and some of these are domain-oriented [7] [8], and thiscannegatively influence other Natural Language Processing applications. Recently, general purpose WSD methods have been investigated with the purpose of integrating these in other applications. We can mention the studies proposed in [9] (focused on nouns) and [10] (focused on verbs).

In this paper, we present an exploratory study of knowledge-based WSD methods (specifically, based on word overlapping, web search, graphs and a method for disambiguation in multi-document scenario) for verbs in BrazilianPortuguese, using WordNet-Pr [11] as sense repository and WordReference®[1] as bilingual dictionary; and a comparison with the results obtained for nouns[9].

The results show that, in All-words evaluation, no methods outperformed the baseline and, in Lexical sample evaluation, the multi-document scenario helped to outperform the baseline. A comparison with the WSD for nouns was performed and the results agreed with the literature, i.e., WSD for verbs is more difficult than WSD for nouns.

The remainder of this paper is structured as follows: Section 2 introduces concepts related to WSD and an overview of the related works for Brazilian Portuguese; the adaption of WSD classic methods for Brazilian Portuguese and the implemented

[1] http://www.wordreference.com/

methods are presented in Section 3; Section 4 shows the performance of the different WSD methods and a comparison between results obtained for verbs and nouns; finally, there are concluding remarks and an outlook of future work in Section 5.

2. CONCEPTS AND RELATED WORK

As we mentioned, WSD aims at selecting the correct sense of a word within a given context using a pre-specified sense repository [2].Basically,WSD methods(a) receive a target word (to disambiguate), a context (words around the target word) and a sense repository (this can be dictionaries, thesaurus, ontologies or wordnets) as input, and (b)execute the automatic disambiguation, showingthe correct sense for the target word as output [1].

The WSD task may be seen in two ways: (1) disambiguating a limited sample of content words in a text, called Lexical sample task; and (2) disambiguating all content words included in a text, called All-words task.

According to the use of resources and techniques, WSD methods can be classified as knowledge-based, corpus-based and hybrid methods [2]. Knowledge-based methods use linguistic resources and similarity measures to disambiguate a wide range of words. This approach is useful for All-words task (because of the use of broad linguistic resources) but the performance obtained by these methods are not so good. Corpus-based methods use sense-annotated corpus to yield machine learning classifiers. This approach is useful for the Lexical sample task (because the number of words to disambiguated is limited by the corpussize) and the performance of these methods is better than the Knowledge-based methods. Finally, hybrid methods use techniques from knowledge and corpus-based methods.

For Portuguese, there are few studies and some of these are domain-oriented. This may negatively influence other Natural Language Processing applications. Recently, General Purpose WSD methods have been proposed. Below, we briefly show some of the main related works for Portuguese that support this investigation.

In [7], a WSD method based on Inductive Logic Programming for Machine Translation task is proposed. Inductive Logic Programming is characterized by using machine learning methods and propositional logic rules. This method was focused on the disambiguation of 10 English verbs with high polysemy (to ask, come, get, give, go, live, look, make, take, and tell) to their respective Portuguese verbs. The author performed some experiments and showed that the proposed method outperformed the most frequent translation method and other methods based on machine learning.

In [8], a geographical disambiguation method for disambiguating place names is presented. This method used an ontology created in this work, called OntoGazetter, as knowledge base. This ontology is composed by place concepts. The results showed that OntoGazetter positively contributes to geographical disambiguation.

The first research on general purpose WSD methods for Brazilian Portuguese is presented in [9]. The authors investigated Knowledge-based WSD methods for common nouns, using WordNet-Pr[10] as sense repository and WordReference® as bilingual dictionary (since the language was Portuguese). In this work, besides the investigation of WSD methods, the author proposed a WSD method based on co-occurrence graphs and a variation of Lesk algorithm [12] for multi-document scenario. The results showed that, although the method does not outperform the baseline (most frequent sense), it contributes to the Word Sense Disambiguation in a multi-document scenario.

In [11], two verb sense disambiguation methods for European Portuguese were developed, using ViPer[13] as sense repository. The proposed methods were based on rules, machine learning and, finally, a combination of the best results of both. The baseline was the most frequent sense method and this was difficult to be outperformed, thus, a combination of methods was performed to outperform the baseline.

3. WORD SENSE DISAMBIGUATION

3.1. *Previous considerations*

A previous step to implement the methods was the choice of the sense repository. For Portuguese, there are some sense repositories as WordNet-Br [14], OpenWordNet-Pt [15] and Onto-pt[16]. For this work, WordNet-Pr 3.0 was chosen (developed for English) as sense repository. This choice was made because of the following reasons: WordNet-Pr is the most used sense repository in the literature; WordNet-Pr is considered a linguistic ontology, thus, it includes concepts and words written in English; and some sense repositories for Portuguese are under development or have a lower coverage than WordNet-Pr.

Another issue to consider is the choice of the WSD methods, because of the need of general purpose WSD methods and its integration with other applications. We chose four Knowledge-based WSD methods, each one from a different approach: using word overlapping [12], web search [17], graphs and similarity measures[18], and, finally, a method that is used in multi-document scenario[9].

After the selection of the WSD methods, we had to adapt these methods (some of these developed for English, initially)for Portuguese, because synsets indexed in WordNet-Pr are written in English. The way to adapt these is the same as used in[9] and it is described as follows: to obtain all synsets for a Portuguese word, we, first obtain all English translations from a bilingual dictionary (in our case, WordReference®), and, then, we obtain all synsetsfor every English translation, using WordNet-Pr. In Figure 1, we can see how this was performed with the verb "*informar*":

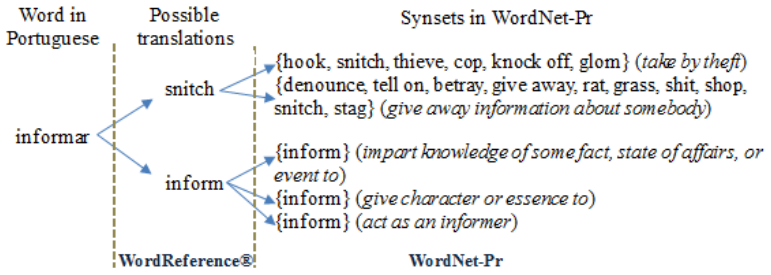


Figure 1. *Method to obtain synsets \for a Brazilian Portuguese verb*

Besides getting synsets, all methods used the following pre-processing steps: (1) sentence splitting; (2) part-of-speech tagging with MXPOST tagger[19]; (3) removal of stopwords; (4) lemmatization of the content words; and (5) target words detection and context representation.

The following subsections describe the WSD Methods investigated in this work.

3.2. *Baseline methods*

In this work, we use 2 methods to compare with the implemented WSD methods. The first of these uses the most frequent sense (MFS) to determine the correct sense of a word. The MFS method uses a sense repository in which the indexed senses for a word are sorted by frequency and, then, it chooses the first sense.

For this work, the way it was adapted is described below: firstly, the MFS method chooses the first translation shown by WordReference® for a Brazilian verb (this is because the results shown by WordReference® are sorted by frequency), and, then, it chooses the first synset in the synset list shown by WordNet-Pr for the selected translation (this is because the results shown by WordNet-Pr are also sorted by frequency).

The second is a random, and blind, method that consists in, firstly, choosing a random translation for a Brazilian verb from the bilingual dictionary and, then, choosing a random synset from the synset list shown by WordNet-Pr for the selected translation.

3.3. *Word overlapping*

The most representative method from this approach is proposed in [12] (called Lesk for practical purposes). This method selects the sense of a word that has more common words with the words in its context window. For this approach, the configurations proposed in [9] were used. This method has six variations: (G-T) using synset glosses of the target word (word to be disambiguated) to compare with labels composed of possible word translations in the context; (S-T) using synset sample sentences of the target word to compare with labels composed of possible word translations in the context; (GS-T) using synset glosses and sample sentences of the target word to compare with labels composed of possible word translations in the context; (S-S) using only synset sample sentence of the target word to compare with labels composed of the sample sentences of all possible synsets for the context words; (G-G) using only synset glosses of the target word to compare with labels composed of the glosses of all possible synsets for the context words; and (GS2) using synset sample sentences and synset glosses of the target word to compare with labels composed of all possible synset sample sentences and glosses for the context words.

Besides these variations, we add other variations by modifying the length and the balance of the context window, this was done because literature says that verbs need unbalanced context windows, having a longer right side in the context window [20]. We use three window variations: (2-2) two words in the left and two words in the right; (1-2) one word in the left and two words in the right; (1-3) one word in the left and three words in the right; and (2-3) two words in the left and three words in the right.'

3.4. *Web search*

The Web Search-based method is the one proposed in [17] (called Mihalcea for practical purposes). This method constructs word pairs in order to disambiguate a word in the context of other word. This method works as follows: for a target word, the nearest random content word is used as context; then, the method

obtains all synsets of the target word; then, queries are constructed, using the combination of every synset with the context, and posted on web search; finally, the synset included in the query with the best result is selected as sense for the target word.

For our case, a word pair consists of the verb under focus and the nearest noun in the sentence. Then, the results for every word pair combination are obtained from web, and, finally, the synset included in the word pair with the best result is selected. For this method, Microsoft Bing® was used for searching the web.

3.5. *Graphs*

The Graph-based WSD method is the one proposed in [18] (called Agirre Soroa for practical purposes). The authors in this work proposed 3 variations based on graphs that use PageRank algorithm [21] to rank the synsets. The first method creates a semantic graph with the synsets of all content words included in a sentence and then executes the PageRank algorithm over the generated graph to rank the synsets. Then, the method selects the highest scored synset for every content word. The second method uses the full WordNet graph and executes the PageRank algorithm over this. In this second method, PageRank algorithm is modified to give priority to synsets of all content words. The third method is similar to the second, but the difference is that this method gives priority to synsets of the context words, excepting the target word (and its synsets) and disambiguates one content word by execution (instead of the other methods that disambiguate all content words by execution). This has the assumption that the synset of the target word must be influenced by the synsets of the words around it. For our study, the last method is used because our focus is to disambiguate verbs only.

3.6. *Multi-document scenario*

The last WSD method is the proposed in [9] (called Nóbrega, for practical purposes). This method is used in multi-document scenario. This method uses a multi-document representation of context and assumes that all the occurrences of a word in a

collection of texts have only one sense based in the corpus (one-sense per discourse heuristic). The multi-document representation of context for a word is built getting the "n" (in our case, we used 3 and 5 words) words that most co-occur with the word to disambiguate(target word) in a window of size "n"(assuming these words are the most related to the target word and help selecting relevant context words and the best synset). After the construction of the context window (obtained from the multi-document representation), the Lesk method is used to disambiguate the target word.

4. EVALUATION

4.1. *The corpus*

The CSTNews corpus[2] [22] [23] was used for evaluating the investigated WSD methods. This is a multi-document corpus composed of 140 texts, extracted from Brazilian news agencies, grouped in 50 collections, where texts of the same collection are about the same topic.

This corpus has sense-annotation for nouns [9] and verbs [24] using the WordNet-Pr as sense repository. In general, 5082 verb instances were annotated. These 5082 instances of verbs represent 844 different verbs with 1047 annotated synsets.

In agreement evaluation, the authors used the Kappa measure [25] and percent agreement among annotators. For percent agreement, the authors calculated the total agreement (when all annotators agreed for verb), partial agreement (when half of the annotators agreed, at least) and no-agreement. Due to the use of sense-repository for English and the use of a bilingual dictionary, the percent agreement was measured for translations, synsets and translation-synset pairs. In Table 1, we present the results of agreement evaluation.

According to the literature, the obtained Kappa values are considered moderate. We may see that the translation agreement shows the highest of the three evaluated items (Translation,

[2] Available at: http://www.icmc.usp.br/~taspardo/sucinto/cstnews.html

Synset and Translation-Synset). This happens because the selection of translations is a simpler task than synset selection. Analyzing the percent agreement, no-agreement is very low, the total agreement is higher and the partial agreement is the highest of the three. One of the reasons for this is that some verbs have a lot of senses and most of them look almost the same. Other reasons are the difficult for identifying participle verbs, complex predicates and the selection of a different English translation to annotate a verb.

Table 1. *Agreement measures computed in [24]*

	Kappa	**Total (%)**	**Partial(%)**	**No-Agreement(%)**
Translation	0.648	48.81	48.50	2.69
Synset	0.509	35.12	58.47	6.41
Translation-synset	0.474	31.73	61.29	6.98

4.2. *Comparison of WSD methods*

For evaluation, WSD methods (described from subsection 3.2 to 3.6) were tested in the CSTNews corpus. Two experiments were performed: the first experiment was to disambiguate all verbs included in the corpus, using all proposed WSD methods (all-words task); and the second experiment was to disambiguate 20 polysemic verbs in the corpus (Lexical sample task).

The measures used to evaluate all WSD methods were: Precision (P), which is the number of correctly classified verbs over the number of verbs classified by the method; Recall (R), number of correctly classified verbs over all verbs in the corpus (3); Coverage (C), number of classified verbs over all the verbs in the corpus; and (4) Accuracy (A), the same as (R), but using MFS method when no classification is found.

Results of All-words experiment are shown in Table 2. As one may see, no WSD method outperformed the MFS method, but all methods out-performed the Random method. The best method was the Nóbrega method, using three words as context and the S-T Lesk variation. The reason for this was the little verb sense variation in a collection of texts. We tested all variations of Lesk method (mentioned in Subsection 3.3) and the best configuration was using an unbalanced window (one word left

and two words right) and the S-T variation. This confirms what the literature says: unbalanced windows help to Verb Sense Disambiguation. Mihalcea method got the worst results. One of the reasons for this is, as mentioned in [29], these senses of a verb change a lot in the presence of different nouns. Other reason is the lack of translations for its noun pair, so, it limits the quantity of verbs to disambiguate, and consequently, its coverage. AgirreSoroa method showed a reasonable result, in comparison with the other developed methods.

Table 2**.** *All-words experiment in CSTNews corpus*

	P (%)	R (%)	C (%)	A (%)
MFS	**49.91**	**47.01**	**94.20**	**-**
Random	10.04	9.46	94.20	-
Lesk-Verbs	40.10	37.69	93.98	37.77
Mihalcea	17.21	14.43	83.87	19.44
AgirreSoroa	28.45	26.80	94.20	26.80
Nóbrega-Verbs	**40.33**	**37.97**	**94.14**	**38.00**

Results of Lexical sample experiment are shown in Table 3.In this experiment, twenty random polysemic verbs (two or more senses in the corpus) were chosen and only the precision measure was computed in order to evaluate the performance of the methods (only the bests by approach). The numbers in bold indicate cases in that the methods performed as well as or better than the MFS method. In general, all WSD methods outperformed the random method. It is obvious, because the random method does not follow some heuristic to select a sense. Analyzing the other methods, it can be seen that Nóbrega method was the best method (P: 35.24%) but this did not outperform the MFS method (P: 36.97%).

One reason for this is the little variation of synsets for a sample word in a collection, i.e., some verbs were annotated in a collection using few synsets (see “F” column and “S” column in Table 3). Another reason is that, despite some verbs have high frequency, these have been annotated mostly with the same sense in a collection. This helps Nóbrega method, because, by using a window context based on the words that more co-occur in a

multi-document scenario, it has a more consistent context and it is able to get a better result. Thus, if the Nóbrega method selects the majority sense for a verb, all verb instances will have the same sense, producing a high precision. The other methods (Lesk variation, Mihalcea and AgirreSoroa) presented results according to the All-words experiment, and the best of the three was Lesk variation, and the worst was Mihalcea.

4.3. *Comparison between morphosyntactic classes*

In Table 4, results of All-words experiment for nouns obtained in [9] are presented. Comparing the results of All-words experiment obtained for verbs (shown in Table 2) and nouns (shown in Table 4), it can be noted that verb is more difficult to disambiguate than noun. In the case of the Lesk method, noun senses disambiguation showed the best performance when this used balanced window (two words for the left and the right side). Unlike nouns, Verb Sense Disambiguation results were obtained when this used unbalanced window (one word for the left side and two words for the right side). Analyzing the content to compare, when this method used the content of the synset glosses, the noun sense disambiguation showed better results (Lesk), but when it used the content of the synset samples, the verb sense disambiguation showed better results. In the case of the Mihalcea method, the difference between verbs and nouns was greater. This occurred because noun senses are more stable in the presence of different verbs, unlike verb senses, which are less stable in the presence of different nouns. In the case of the Nóbrega methods, the best configuration for nouns (using the G-T variation) showed better performance than the best for verbs. This confirms that nouns have less meaning variation in the corpus than verbs [26].

5. FINAL REMARKS

In this work, the first exploratory study of classic WSD methods adapted for verbs in Brazilian Portuguese was presented. Due to the need of WSD methods that can be used in different contexts,

knowledge-based WSD methods were chosen. The approaches for knowledge-based WSD methods were: word overlapping, web search, graphs and a method focused on multi-document scenario. Then, we used a journalistic corpus, which included various domains to guarantee the general use, to test these methods.

Table 3. *Lexical sample experiment in CSTNews corpus (F: Frequency; S: Number of synsets; MFS: Most frequent sense Method; R: Random Method; L: Lesk method; M: Mihalcea method; AS: AgirreSoroa method; N: Nóbrega method)*

Word	**F**	**S**	**MFS**	**R**	**L**	**M**	**AS**	**N**
Estragar (ruin)	2	2	0.00	0.00	0.00	0.00	0.00	0.00
Olhar (look)	2	2	0.00	0.00	0.00	0.00	0.00	0.00
Perceber (perceive)	2	2	0.00	0.00	0.00	0.00	0.00	0.00
Gostar (like)	2	2	0.00	100.00	0.00	0.00	0.00	0.00
Exibir (exhibit)	3	2	0.00	50.00	0.00	0.00	0.00	0.00
Resultar (result)	3	3	100.00	0.00	0.00	0.00	0.00	**100.00**
Pertencer (belong)	4	2	100.00	33.33	66.67	0.00	0.00	**100.00**
Voar (fly)	4	2	33.33	0.00	**33.33**	0.00	**33.33**	**33.33**
Entender (understand)	4	3	50.00	0.00	**50.00**	**50.00**	**50.00**	**50.00**
Descobrir (discover)	4	3	50.00	0.00	**50.00**	0.00	**50.00**	**50.00**
Destacar (feature)	5	2	0.00	0.00	0.00	0.00	0.00	0.00
Achar(find)	6	3	0.00	0.00	0.00	0.00	0.00	0.00
Recuperar (recover)	6	4	50.00	25.00	**50.00**	25.00	25.00	**50.00**
Retirar (withdraw)	9	3	100.00	0.00	**100.00**	16.67	0.00	**100.00**
Comandar (command)	12	4	33.33	22.22	**33.33**	28.57	22.22	**33.33**
Marcar (mark)	17	7	0.00	0.00	**20.00**	0.00	**20.00**	0.00
Entrar (enter)	21	4	62.50	0.00	**62.50**	28.57	0.00	50.00
Receber (receive)	36	9	77.78	0.00	50.00	14.29	0.00	55.56
Deixar (leave)	49	16	11.11	0.00	7.41	0.00	**11.11**	**11.11**
Informar (inform)	55	2	71.43	10.71	64.29	0.00	**71.43**	**71.43**
AvgPrecision	-	-	**36.97**	12.06	29.38	8.15	14.15	35.24

Two experiments were performed: All-words and Lexical sample task. Both All-words and Lexical sample tasks showed that no method outperformed the MFS method. However, considering the implemented methods, the Nóbrega method got the best results. One reason for this is the little variation of verb senses for the sample in a collection of texts. A third experiment was

performed, aiming at comparing the performance between morphosyntactic classes (nouns and verbs). The results are consistent with what other studies claim that verbs are more difficult to disambiguate.

In spite of the fact that Wordnet-Pr is a wide resource used in WSD, the tested methods showed some problems with lexical gaps. For instance, the verb "*pedalar*" (action of doing a specific dribble) in the sentence "*O Robinhopedalou*" has not a respective synset in WordNet-Pr. To resolve this problem, a generalization of the Portuguese verbis necessary, using the verb "dribble" ("*driblar*", in Portuguese).

As we could see, there is still room for improvements in WSD for verbs. A future work is the use of repositories focused on verbs (and developed for Portuguese), which contain syntactic and semantic information that might be added to the methods to improve their performance.

Table 4. *All-words experiment for nouns*

Method	**P (%)**	**R (%)**	**C (%)**	**A (%)**
MFS	51.00	51.00	100.00	-
Lesk-Noun	42.20	41.20	91.10	41.20
Mihalcea	39.71	39.47	99.41	39.59
Nóbrega-Noun	49.56	43.90	88.59	43.90

REFERENCES

1. Jurafsky, D. & Martin, J. H. 2009. Speech and language processing: An introduction to natural language processing. *Speech Recognition, and Computational Linguistics*. 2nd edition. Prentice-Hall.
2. Agirre, E. & Edmonds, P. 2006. Introduction. *Word Sense Disambiguation: Algorithms and Applications* (pp. 1-28). Springer.
3. Navigli, R. 2009. Word sense disambiguation: A survey. In *ACM Computational Survey***,** 41, 1-69.
4. Gao, N., Zuo, W., Dai, Y. & Wei, L. 2014. Word sense disambiguation using WordNet semantic knowledge. In proceedings of the *Eighth International Conference on Intelligent Systems and Knowledge Engineering*, (pp. 147-156), Springer Berlin Heidelberg, China.

5. Fillmore, C. J. 1968. The case for case. In *Universals in Linguistic Theory* (pp. 1-89). New York.
6. Che, W. & Liu, T. 2010. Jointly modeling wsd and srl with markov logic. In proceedings of *23rd International Conference on Computational Linguistics*, (pp. 161-169), Association for Computational Linguistics, China.
7. Specia, L. 2007. Uma Abordagem Híbrida Relacional para a Desambiguação Lexical de Sentido na Tradução Automática. PhD thesis, Instituto de Ciências Matemáticase de Computação- ICMC-USP.Brazil.
8. Machado, I. M., de Alencar, R.O., de Oliveira Campos Junior, R. & Davis, C. A. 2011. An ontological gazetteer and its application for place name disambiguation in text. In *Journal of the Brazilian Computer Society*, 17, 267-279.
9. Nóbrega, F. A. A. & Pardo, T. A. S. 2014. General purpose word sense disambiguation methods for nouns in Portuguese. In proceedings of the *11th International Conference on Computational Processing of Portuguese* (pp. 94-101), Brazil.
10. Travanca, T. 2013. Verb sense disambiguation. MSc thesis. *Instituto Superior Técnico*. Universidade Técnica de Lisboa. Portugal.
11. Fellbaum, C. 1998. *WordNet, An Eletronic Lexical Database*. MIT Press.
12. Lesk, M. 1986. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In proceedings of *5th Annual International Conference on Systems Documentation* (pp. 24-26), Association for Computing Machinery, USA.
13. Baptista, J. 2012. ViPEr: A lexicon-grammar of European Portuguese verbs. In proceedings of the *31st International Conference on Lexis and Grammar* (pp. 10-16), Czech Republic.
14. Dias da Silva, B. C. 2005. A construção da base da wordnet.br : Conquistas e desafios. In proceedings of *XXV Congresso da Sociedade Brasileira de Computação*.
15. Paiva, V., Rademaker, A. & Melo, G. 2012. OpenWordNet-PT: An open Brazilian Wordnet for reasoning. In proceedings of *COLING 2012: Demonstration Papers* (pp. 353-360), India.
16. Gonçalo Oliveira, H., Antón, L. P. & Gomes, P. 2012. Integrating lexical-semantic knowledge to build a public lexical ontology for Portuguese. In *Natural Language Processing and Information Systems, Proceedings of 17th International Conference on*

Applications of Natural Language to Information Systems (pp. 210-215), The Netherlands,

17. Mihalcea, R. & Moldovan, D. I. 1999. A method for word sense disambiguation of unrestricted text. In proceedings of *37th Annual Meeting of the Association for Computational Linguistics* (pp. 152-158), USA.
18. Agirre, E. & Soroa, A. 2009. Personalizing pagerank for word sense disambiguation. In proceedings of the *12th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 33-41), *Association for Computational Linguistics*.
19. Ratnaparkhi, A. 1996. A maximum entropy model for part-of-speechtagging. In proceedings of the *First Empirical Methods in NLP Conference* (pp. 133-142), Association for Computational Linguistics.
20. Vasilescu, F., Langlais, P. & Lapalme, G. 2004. Evaluating variants of the lesk approach for disambiguating words. In proceedings of *Language Resources and Evaluation (LREC 2004)* (pp. 633-636), Portugal.
21. Brin, S. & Page, L. 1998. The anatomy of a large-scale hypertextual web search engine. In proceedings of *17th International World-Wide Web Conference* (WWW 1998).
22. Aleixo, P. & Pardo, T. A. S. 2008. CSTNews: Um córpus de textos jornalísticos anotados segundo a teoria discursiva multidocumento CST (cross-documentstructuretheory). Relatório Técnico 326, Instituto de Ciências Matemáticas e de Computação., Universidade de São Paulo. Brazil.
23. Cardoso, P. C. F., Maziero, E. G., Jorge, M. L. R. C., Seno, E. M. R., Felippo, A. D., Rino, L. H. M., Das Graças, M. V. N. & Pardo, T. A. S. 2008. CSTNews – a discourse-annotated corpus for single and multi-document summarization of News texts in Brazilian portuguese. In *proceedings of III Workshop "A RST e os Estudos do Texto,"* (pp. 88-105), Sociedade Brasileira de Computação, Brazil.
24. Sobrevilla Cabezudo, M. A., Maziero, E.G., Souza, J. W. C., Dias, M. S., Cardoso, P. C. F., Balage Filho, P. P., Agostini, V., Nóbrega, F. A. A., Barros, C. D., Di Felippo, A. & Pardo, T. A. S. 2014. Anotação de Sentidos de Verbos em Notícias Jornalísticas em Português do Brasil. In proceedings of the *XII Encontro de Linguística de Corpus*-ELC, Brazil.
25. Carletta, J. 1996. Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22, 249-254.

26. Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K. J. 1990. Introduction to Wordnet: An on-line lexical database. In *International Journal of Lexicography*, 3, 235-244.

ACKNOWLEDGMENT

Part of the results presented in this paper were obtained through research on a project titled "Semantic Processing of Texts in Brazilian Portuguese", sponsored by Samsung Eletrônica da Amazônia Ltda. under the terms of Brazilian federal law No.8.248/91.

MARCO ANTONIO SOBREVILLA CABEZUDO
NÚCLEO INTERINSTITUCIONAL
DE LINGUÍSTICA COMPUTACIONAL,
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO,
UNIVERSIDADE DE SÃO PAULO
AV. TRABALHADOR SÃO-CARLENSE,
400 - CENTRO CEP: 13566-590 –
SÃO CARLOS – SÃO PAULO, BRASIL.
E-MAIL: <MARCOSBC@ICMC.USP.BR>

THIAGO ALEXANDRE SALGUEIRO PARDO
NÚCLEO INTERINSTITUCIONAL
DE LINGUÍSTICA COMPUTACIONAL,
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO,
UNIVERSIDADE DE SÃO PAULO,
AV. TRABALHADOR SÃO-CARLENSE,
400 - CENTRO CEP: 13566-590 –
SÃO CARLOS – SÃO PAULO, BRASIL.
E-MAIL: <TASPARDO@ICMC.USP.BR>

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 149–172
Received 22/02/2015, Accepted 18/05/2015, Final 14/06/2015.
ISSN 0976-0962, http://ijcla.bahripublications.com

Computing Efficiently the Closeness of Word Sets in Natural Language Texts

DOMENICO CANTONE
SALVATORE CRISTOFARO
GIUSEPPE PAPPALARDO
Universita di Catania, Italy

ABSTRACT

We consider some search problems which have applications in statistical text analysis and natural language processing. Given two sets of words A *and* B*, we propose a statistical, corpus-based measure of the "closeness" between* A *and* B *in texts. Our proposed measure involves the search, throughout a text corpus, of the words in* A *and* B*, under the restriction that these words should co-occur within a given maximum distance n. We address the problem of efficiently computing this closeness measure and present algorithms for it.*

Keywords: Natural language processing, text search algorithms, closeness measure

1. INTRODUCTION

Discovering and retrieving relations between words is a central topic in computational linguistics, text analysis, and information retrieval. This is particularly true, for instance, in the case of the (semantic) similarity relation between words – *word similarity* – which has several important applications in natural language processing (see [1]). Various notions of word similarity have been proposed and studied over the years. Usually, word similarity relations are supplied with a measure – *a similarity measure* – that provides a statistical estimate of the degree of

similarity among words, i.e., how much a word is related or similar to another word.

A conceptually simple case of similarity relation on words is provided by the following notion of *word closeness*: a word x is close to another word y if ("most of the times") x co-occurs with y within a given maximum distance n in a given collection of texts. This notion of word closeness is at the basis of various statistical measures of similarity that use the so-called corpus-based approach [2-5]. In the corpus-based model, similarity measures are computed using co-occurrence statistics of the words from a large corpus of collected texts in which the words occur. Note that this is different from the knowledge-based approach, which includes graph-based algorithms operating on lexical databases such as the WordNet [6-9].

A corpus-based measure for the above relation of word closeness can be easily provided by dividing the number of times that x co-occurs with y in the corpus within a distance of at most n, by the number of times that x occurs in the corpus. The higher is this ratio, the closer is x to y.

We generalize this relation of word closeness to sets of words (relative to a corpus); accordingly, a set A of words is close to another set B of words if the words of A co-occur with the words of B within a given maximum distance n in the corpus. We can define a statistical measure for this relation as follows. We assume that the corpus is provided as a finite ordered sequence (i.e., a string) of words. A *window* is a sequence of at most n consecutive words of the corpus. A window containing the set A is minimal if it cannot be shortened without skipping any word in A. Then, we count the number of minimal windows containing A that can be extended to windows containing also B, and divide this number by the number of minimal windows containing A. The resulting ratio is the *closeness factor of A to B* (for a formal statement, see *Problem* 1 in Section 3.) Again, the higher is the closeness factor, the closer is A to B.

The closeness factor of word sets has applications in the field of *text simplification* [10], a subfield of natural language processing concerned with the problem of reducing the syntactic

complexity of a text, while retaining its information content and meaning. The basic idea here is that a text can be simplified by repeatedly removing from it one or more "semantically superfluous" words, i.e., words that do not affect its whole meaning. Note that simplifications of this kind can be used, for instance, to make it easier to analyze the meaning of texts in such contexts as *sentiment analysis* [11], concerned with the study of opinions expressed in text documents. Then, a possible approach to text simplification based on the closeness factor of word sets, could be the following one. Let us first introduce a useful notation to ease our presentation: for a set of words S, we denote by Sym (S) the set of words which are similar to the words in S, under a given relation of word similarity. Next, let T be the set of the words occurring in the text we intend to simplify. After choosing the first candidate word $w_1 \in T$ to be removed, we form the set Sym(T \ $\{w_1\}$) to Sym(T); and compute the closeness factor of Sym(T \ $\{w_1\}$) to Sym(T); if this is greater than or equal to some fixed threshold $t \leq 1$, we "safely" remove the word w_1 from T. We then select a second candidate word $w_2 \in T \setminus \{w_1\}$ to be removed from $T \setminus \{w_1\}$ and proceed as above. We keep removing words form T as long as the closeness factor remains above the threshold t. Let $w_1, w_2, ..., w_r$ be the list of the words that are removed during this process. Then, the text is simplified by eliminating from it the words of the set Sym($\{w_1, w_2, ..., w_r\}$). (See Example 1 at the end of Section 3.1.) The "simplification level", i.e., the number of words that are actually removed, depends on the particular candidates selected at each step. The goal is to obtain as high a simplification level as possible. Needless to say, trying all possible choices of the candidate words to be removed is far too expensive, since their number is exponential. Thus, it is essential to develop good selection heuristics, and make the choice of the candidates on the basis of them. In any case, since each step involves the calculation of a closeness factor, it is of prime importance to first design fast algorithms for performing such calculations efficiently.

In this paper we address the latter problem and present some efficient algorithms for computing the closeness factor of two sets of words.

The scenario sketched above for the simplification of a text reduces to the following optimization problem: given a set S of words and a threshold $t \leq 1$, determine a subsets S' of S, of minimum size, such that the closeness factor of S' to S is greater than or equal to t. However, here we will not be concerned with this problem, as it will be the topic of future investigations.

The paper is organized as follows. In Section 2 we introduce basic notations and terminology. Then, in Section 3, we formally define the main problem we are interested in, namely the computation of the closeness factor of two sets of words. In particular, we first develop efficient algorithms for two simpler problems related to it and then combine them into an efficient solution to our main problem (cf. Section 3.1). Finally, in Section 4, we draw our conclusions.

2. BASIC NOTATIONS

We will deal with strings of *words*, i.e., finite ordered sequences (possibly empty) of word occurrences. The *empty string* is denoted with ε and the length of a string σ is denoted with $|\sigma|$. Given a string σ and a word ω, we write $\sigma.\omega$ for the string obtained by adding the word ω to the end of σ. Note that, for each word ω, $\varepsilon.\omega$ coincides with the string of length 1 consisting in a single occurrence of ω. It is also convenient to assume the existence of a special word, the *null-word*, denoted with Λ.

Let σ be a string (of words). Given an index $0 \leq i < |\sigma|$, we denote with σ_i the $(i + 1)$st word of σ (from left to right); we extend this notation by also putting $\sigma_i =_{\text{Def}} \Lambda$ for any $i < 0$ or $i \geq |\sigma|$. Given two integers h and k, we denote with $[h, k]$ the set (interval) of integers ℓ such that $h \leq l \leq k$. The *substring of σ between positions h and k*, where h and k are integers, is the string $\sigma_{[h, k]}$ defined recursively by

$$\sigma_{[h,k]} =_{Def} \begin{cases} \varepsilon, & \text{if } h > k \\ \sigma_{[h,k-1]} \cdot \sigma_k, & \text{otherwise.} \end{cases}$$

The set $\{\sigma_i : 0 \leq i < |\sigma|\}$ of the words occurring in σ is briefly denoted by $\langle\sigma\rangle$. Given two integers i and n, with $n \geq 1$, we call $\sigma_{[i,\ i+n-1]}$ the *window of σ of length n (starting) at position i.* The window $\sigma_{[i,i+n-1]}$ is a *minimal window containing a set S of words*, if $S \subseteq \langle\sigma_{[i,i+n-1]}\rangle$ but $S \not\subseteq \langle\sigma_{[h,k]}\rangle$, for all positions $i \leq h$, $k \leq i + n - 1$ such that the interval $[h, k]$ is strictly contained in the interval $[i,\ i + n - 1]$. A *text* is a nonempty string containing no occurrence of the null-word Λ, i.e., $\Lambda \notin \langle\tau\rangle$. The cardinality of a (finite) set S of words is denoted with $|S|$. In the sequel, by a set of words we shall always mean a nonempty set of words not containing the null-word Λ.

3. FORMALIZING THE PROBLEM: SOME PRELIMINARY ALGORITHMS

In this section and in the subsequent one we address in details the problem of computing the closeness factor introduced in Section 1.

We begin with the following formal definitions.

Definition 1. *Let τ be a text, S a set of words, and n a window-length. For each integer i we put*:

$$\mathcal{M}_{\tau,S,n}(i) =_{Def} \min(\{k \in [i-1,\ i+n-1] : S \cap \langle\tau_{[i,\ k]}\rangle = S \cap \langle\tau_{[i,\ i+n-1]}\rangle\})$$

$$K_{\tau,S,n}(i) =_{Def} |S \cap \langle\tau_{[i,\ i+n-1]}\rangle|,$$

$$\mathcal{E}_{\tau,S,n}(i) =_{Def} \max(\{k \in [i-n,+1,\ i] : S \subseteq \langle\tau_{[k,\ k+n-1]}\rangle\} \cup \{i-n\}).$$

(*Thus if $S \subseteq \langle\tau_{[i,\ i+n-1]}\rangle$ and $\tau_i \in S \setminus \langle\tau_{[i,i+1,\ \mathcal{M}_{\tau,S,n(i)}]}\rangle$, the window $\tau_{[i\ ,\ \mathcal{M}_{\tau,S,n(i)}]}$ is a minimal window containing S; moreover, $S \subseteq \langle\tau_{[i,\ i+n-1]}\rangle$ iff $K_{\tau,S,n(i)} = |S|$. In addition, for $j \in [i,\ i+n-1]$, the window $\tau_{[i,\ j]}$ can be extended to a window of length n containing S, iff $j - \mathcal{E}_{\tau,S,n}(i) < n$.*)

Note that, for each $i \leq -n$, we have that $\mathcal{M}_{\tau,S,n}(i) = i - 1$, $\mathcal{K}_{\tau,S,n}(i) = 0$, and $\mathcal{E}_{\tau,S,n}(i) = i - n$; moreover, for $i > -n$, we have

$$\mathcal{E}_{\tau,S,n}(i) = \begin{cases} i, & \text{if } \mathcal{K}_{\tau,B,n}(i) = |B| \\ \max(\{n - i, \mathcal{E}_{\tau,B,n}(i-1)\}), & \text{otherwise.} \end{cases} \tag{1}$$

Also, observe that, for each integer i, we have that $\mathcal{M}_{\tau,S,n}(i) = \mathcal{M}_{\tau_{[0,i+n-1]},S,n}(i)$, $\mathcal{K}_{\tau,S,n}(i) = \mathcal{K}_{\tau_{[0,i+n-1]},S,n}(i)$, and $\mathcal{E}_{\tau,S,n}(i) = \mathcal{E}_{\tau_{[0,i+n-1]},S,n}(i)$.

Next, we state formally the main problem we are interested in.

Problem 1. Given a text τ,[1] two sets A and B of words, and a window-length n, compute the ratio

$$(A \ltimes B)_{\tau,n} =_{Def} \frac{\Omega^{+}_{\tau,A,n}(B)}{\Omega^{+}_{\tau,A,n}(B) + \Omega^{-}_{\tau,A,n}(B)},$$

where

$$\Omega^{+}_{\tau,A,n}(B) =_{Def} \Big|\{i \in [0, |\tau| - 1] : \mathcal{K}_{\tau,A,n}(i) = |A|, \tau_i \in A \setminus \langle \tau_{[i+1,\mathcal{M}_{\tau,A,n}(i)]} \rangle \text{ AND } (\mathcal{M}_{\tau,A,n}(i) - \mathcal{E}_{\tau,B,n}(i) < n)\}\Big|$$

and

$$\Omega^{-}_{\tau,A,n}(B) =_{Def} \Big|\{i \in [0, |\tau| - 1] : \mathcal{K}_{\tau,A,n}(i) = |A|, \tau_i \in A \setminus \langle \tau_{[i+1,\mathcal{M}_{\tau,A,n}(i)]} \rangle \text{ AND } (\mathcal{M}_{\tau,A,n}(i) - \mathcal{E}_{\tau,B,n}(i) \geq n)\}\Big|.$$

[1] Here τ stands for the corpus where the (co-)occurrences of the words of A and B are searched for (see the discussion in Section 1).

We call $(A \ltimes B)_{\tau,n}$ the *closeness factor of A to B relative to τ and n*.

Remark 1. Note that $\Omega^{+}_{\tau,A,n}(B)$ is equal to the number of the windows $\tau_{[i,i+m-1]}$ of τ, with $0 \leq i < |\tau|$ and $m \leq n$, such that (a) $\tau_{[i,i+m-1]}$ is a minimal window containing A, and (b) there exists $h \in [i+m-n, i]$ such that $B \subseteq \langle \tau_{[h,h+n-1]} \rangle$ (i.e., $\tau_{[i,i+m-1]}$ can be extended to a window of length n containing B, namely the window $\tau_{[h,h+n-1]}$). Moreover, $\Omega^{+}_{\tau,A,n}(B) + \Omega^{-}_{\tau,A,n}(B)$ is equal to the number of the minimal windows of τ containing A.

We will solve Problem 1 by considering separately the following two problems whose solutions can be easily combined to produce a solution to our main problem, as shown in Section 3.1.

Problem 2. Given a text τ, a set S of words, and a window-length n, determine the value $\mathcal{M}_{\tau,S,n}(i)$, for each position i of τ.

Problem 3. Given a text τ, a set S of words, and a window-length n, determine the value $\mathcal{K}_{\tau,S,n}(i)$, for each position i of τ.

Let us first consider Problem 2. We shall first present a basic algorithm for it, namely the algorithm Algo1 reported in Fig. 2; this will be subsequently refined into a more efficient variant named Algo2, reported in Fig. 3. We begin with the following observations.

Firstly, note that the value $\mathcal{M}_{\tau,S,n}(i)$ does not depend on the word τ_{i+n-1}, i.e., the last word of the window $\tau_{[i,i+n-1]}$, provided that such word is not contained in the set S. Thus, in particular, if $\tau_{i+n-1} \notin S$, we have that $\mathcal{M}_{\tau,S,n}(i) = \mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i)$.[2] Secondly, if $\tau_{i+n-1} \in S$, we can easily compute $\mathcal{M}_{\tau,S,n}(i)$ by using the value $\mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i)$ and the number $\mathcal{C}$ of occurrences of the word τ_{i+n-1} in the window $\tau_{[i,i+n-2]}$; indeed, if $\mathcal{C} = 0$ then $\tau_{[i,i+n-2]}$ contains no occurrences of τ_{i+n-1} and, in this case, we

[2] Since $\Lambda \notin S$, we have $\mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i) = \mathcal{M}_{\widetilde{\tau},S,n}(i)$, for $0 \leq i < |\tau| - n + 1$, where $\widetilde{\tau}$ is the string obtained by replacing the word in τ at position $i+n-1$ by the null-word Λ.

plainly have that $\mathcal{M}_{\tau,S,n}(i) = i + n - 1$; otherwise, if $\mathcal{C} \neq 0$, we plainly have that $\mathcal{M}_{\tau,S,n}(i) = \mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i)$. The number $\mathcal{C}$ can be easily expressed in terms of the number $\mathcal{C}'$ of occurrences of τ_{i+n-1} in the window $\tau_{[i-1,i+n-2]}$ by means of the following simple formula:

$$\mathcal{C} = \begin{cases} \mathcal{C}', & \text{if } \tau_{i-1} \neq \tau_{i+n-1} \\ \mathcal{C}' - 1, & \text{otherwise.} \end{cases}$$

Thus, the value of $\mathcal{M}_{\tau,S,n}(i)$ can be easily computed from the values $\mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i)$ and $\mathcal{C}'$.

Next, we turn our attention to the computation of the value $\mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i)$. We show how to compute it from $\mathcal{M}_{\tau,S,n}(i-1)$, using the positions of the occurrences of τ_{i-1} in the window $\tau_{[i-1,i+n-2]}$. Suppose first that $\tau_{i-1} \in S$. Let $\mathcal{L}$ be the set of the positions k of τ such that $\tau_k = \tau_{i-1}$, where $\mathcal{M}_{\tau,S,n}(i-1) < k < i + n - 1$.[3] Moreover, let $\mathcal{C}''$ be the number of occurrences of the word τ_{i-1} in the window $\tau_{[i-1,i+n-2]}$. Then $\mathcal{C}'' - 1 - |\mathcal{L}|$ is equal to the number of occurrences of the word τ_{i-1} which are strictly comprised between positions $i-1$ and $\mathcal{M}_{\tau,S,n}(i-1)$ of τ; i.e.,

$$\mathcal{C}'' - 1 - |\mathcal{L}| = \left|\{k \in [i, \mathcal{M}_{\tau,S,n}(i-1) - 1] : \tau_k = \tau_{i-1}\}\right|.$$

We observe the following facts. (A) If $\mathcal{C}'' - 1 - |\mathcal{L}| \neq 0$, there are occurrences of τ_{i-1} between positions $i-1$ and $\mathcal{M}_{\tau,S,n}(i-1)$ of τ, so that the value $\mathcal{M}_{\tau,S,n}(i-1)$ does not depend by any means on the word τ_{i-1}; hence, in this case, we have that $\mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i) = \mathcal{M}_{\tau,S,n}(i-1)$. Similarly, (B) if $\mathcal{L} = \emptyset$, then $\mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i) = \mathcal{M}_{\tau,S,n}(i-1)$ as well. When (C) $\mathcal{C}'' - 1 - |\mathcal{L}| = 0$ and $\mathcal{L} \neq \emptyset$ hold, it can be readily verified that, instead, $\mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i) = \min(\mathcal{L})$. (See Fig. 1 for a pictorial illustration of cases (A), (B), and (C) above.)

Finally, observe that if $\tau_{i-1} \notin S$, then $\mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i) = \max(\{i-1, \mathcal{M}_{\tau,S,n}(i-1)\})$.

[3] Plainly, since $\tau_{i-1} \in S$, we have $\mathcal{M}_{\tau,S,n}(i-1) \geq i-1$. Moreover, $\mathcal{M}_{\tau,S,n}(i-1) = i-1$ iff $S = \{\tau_{i-1}\}$, and if $\mathcal{M}_{\tau,S,n}(i-1) > i-1$, then $\tau_{\mathcal{M}_{\tau,S,n}(i-1)} \neq \tau_{i-1}$.

The previous observations can be stated formally as Lemmas 1 and 2 below. These will be expressed in terms of some maps to be introduced in Definition 2 below. In fact, rather than computing directly the values $\mathcal{M}_{\tau,S,n}(i)$, it is more convenient to compute the quantities $\mathcal{M}_{\tau,S,n}(i)$ defined below as this allows one to avoid dealing with the case $\tau_{i-1} \notin S$ examined above, yielding simpler calculations. Note in fact that, for each i, $\mathcal{M}_{\tau,S,n}(i) = \max\{\mathcal{M}_{\tau,S,n}(i), i-1\}$.

Definition 2. *For each text τ, set S of words, and window-length n, word w, we define the following maps:*

$$\mathcal{M}_{\tau,S,n}(i) =_{Def} \begin{cases} \mathcal{M}_{\tau,S,n}(i-1), & \text{if } i > -n \text{ AND} \\ & \qquad \mathcal{M}_{\tau,S,n}(i) = i-1 \\ \mathcal{M}_{\tau,S,n}(i), & \text{otherwise} \end{cases}$$

$$\mathcal{C}^w_{\tau,n}(i) =_{Def} |\{k \in [i, i+n-1] : \tau_k = w\}|$$

$$\mathcal{L}^w_{\tau,S,n}(i) =_{Def} \{k \in [i, i+n-1] : k > \mathcal{M}_{\tau,S,n}(i) \text{ AND } \tau_k = w\}.$$

Below we let τ be a fixed text, S be a fixed set of words, and n be a fixed window-length. In addition, to simplify the notations, for each integer i and word w, we shall write $\mathcal{M}(i)$, $\mathcal{C}^w(i)$, and $\mathcal{L}^w(i)$ for the values $\mathcal{M}_{\tau,S,n}(i)$, $\mathcal{C}^w_{\tau,n}(i)$, and $\mathcal{L}^w_{\tau,S,n}(i)$, respectively; also, we shall write $\widetilde{\mathcal{M}}(i)$, $\widetilde{\mathcal{L}}^w(i)$, and $\widetilde{\mathcal{C}}^w(i)$ for the values $\mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i)$, $\mathcal{L}^w_{\tau_{[0,i+n-2]},S,n}(i)$, and $\mathcal{C}^w_{\tau_{[0,i+n-2]},n}(i)$, respectively.[4]

We are now ready to state the two lemmas which summarize our previous discussion.

Lemma 1. *For each i, the following properties hold:*

(a) If $\tau_{i-1} \notin S$, then $\widetilde{\mathcal{M}}(i) = \mathcal{M}(i-1)$, $\widetilde{\mathcal{L}}^w(i) = \mathcal{L}^w(i-1)$, and $\widetilde{\mathcal{C}}^w(i) = \mathcal{C}^w(i-1)$, for each $w \in S$.

[4] Observe that, if $w \neq \Lambda$ and $0 \leq i < |\tau| - n + 1$, then $\mathcal{C}^w_{\tau_{[0,i+n-2]},n}(i) = \mathcal{C}^w_{\widetilde{\tau},n}(i)$, where $\widetilde{\tau}$ stands for the string obtained by replacing the word at position $i+n-1$ of τ by the null-word Λ; similarly for $\mathcal{L}^w_{\tau_{[0,i+n-2]},S,n}(i)$. (See also footnote 2.)

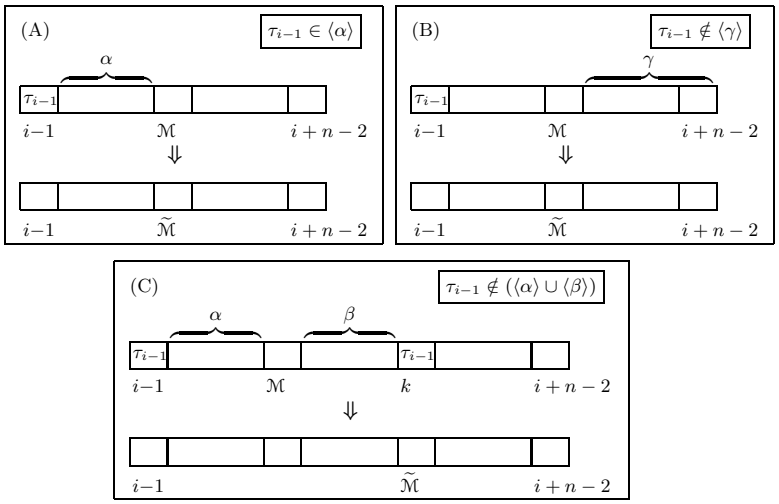


Fig. 1. The computation of $\mathcal{M}_{\tau_{[0,i+n-2]},S,n}(i)$ (denoted as $\widetilde{\mathcal{M}}$) from $\mathcal{M}_{\tau,S,n}(i-1)$ (denoted as $\mathcal{M}$), when $\tau_{i-1} \in S$. In the pictures, α, β and γ denote the windows $\tau_{[i,\mathcal{M}-1]}$, $\tau_{[\mathcal{M}+1,k-1]}$ and $\tau_{[\mathcal{M}+1,i+n-2]}$, respectively, with $k = \min(\mathcal{L})$.
Cases (A) $\mathcal{C}'' - 1 - |\mathcal{L}| \neq 0$; (B) $\mathcal{L} = \emptyset$; and (C) $\mathcal{C}'' - 1 - |\mathcal{L}| = 0$, with $\mathcal{L} \neq \emptyset$. In both cases (A) and (B) we have $\widetilde{\mathcal{M}} = \mathcal{M}$, whereas in case (C) we have $\widetilde{\mathcal{M}} = \min(\mathcal{L})$.

(b) If $\tau_{i-1} \in S$, then
- *(b.1) $\widetilde{\mathcal{C}}^w(i) = \mathcal{C}^w(i-1)$, for each $w \in S \setminus \{\tau_{i-1}\}$;*
- *(b.2) $\widetilde{\mathcal{C}}^{\tau_{i-1}}(i) = \mathcal{C}^{\tau_{i-1}}(i-1) - 1$;*
- *(b.3) if $\mathcal{L}^{\tau_{i-1}}(i-1) \neq \emptyset$ and $\mathcal{C}^{\tau_{i-1}}(i-1) - 1 - |\mathcal{L}^{\tau_{i-1}}(i-1)| = 0$, then $\widetilde{\mathcal{M}}(i) = \min(\mathcal{L}^{\tau_{i-1}}(i-1))$ and $\widetilde{\mathcal{L}}^w(i) = \{j \in \mathcal{L}^{\tau_{i-1}}(i-1) : j > \widetilde{\mathcal{M}}(i)\}$, for each $w \in S$;*
- *(b.4) if $\mathcal{L}^{\tau_{i-1}}(i-1) = \emptyset$ or $\mathcal{C}^{\tau_{i-1}}(i-1) - 1 - |\mathcal{L}^{\tau_{i-1}}(i-1)| \neq 0$, then $\widetilde{\mathcal{M}}(i) = \mathcal{M}(i-1)$ and $\widetilde{\mathcal{L}}^w(i) = \mathcal{L}^w(i-1)$, for each $w \in S$.*

Lemma 2. *For each i, the following properties hold:*

(a) If $\tau_{i+n-1} \notin S$, then $\mathcal{M}(i) = \widetilde{\mathcal{M}}(i)$, $\mathcal{L}^w(i) = \widetilde{\mathcal{L}}^w(i)$, and $\mathcal{C}^w(i) = \widetilde{\mathcal{C}}^w(i)$, for each $w \in S$.
(b) If $\tau_{i+n-1} \in S$, then

(b.1) $\mathcal{C}^{\tau_{i+n-1}}(i) = \widetilde{\mathcal{C}}^{\tau_{i+n-1}}(i) + 1$*;*
(b.2) $\mathcal{C}^{w}(i) = \widetilde{\mathcal{C}}^{w}(i)$*, for each* $w \in S \setminus \{\tau_{i+n-1}\}$*;*
(b.3) if $\widetilde{\mathcal{C}}^{\tau_{i+n-1}}(i) = 0$*, then* $\mathcal{M}(i) = i+n-1$ *and* $\mathcal{L}^{w}(i) = \emptyset$*, for each* $w \in S$*;*
(b.4) if $\widetilde{\mathcal{C}}^{\tau_{i+n-1}}(i) \neq 0$*, then* $\mathcal{M}(i) = \widetilde{\mathcal{M}}(i)$*,* $\mathcal{L}^{w}(i) = \widetilde{\mathcal{L}}^{w}(i)$ *for each* $w \in S \setminus \{\tau_{i+n-1}\}$*, and also* $\mathcal{L}^{\tau_{i+n-1}}(i) = \widetilde{\mathcal{L}}^{\tau_{i+n-1}}(i) \cup \{i+n-1\}$.

Lemmas 1 and 2 readily lead to the basic algorithm Algo1 for Problem 2 reported in Fig. 2. Algorithm Algo1 scans the text τ from left to right. It moves iteratively a window of length n along τ, advancing it one position at a time. During iteration i, the items $\mathcal{M}(i)$, $\mathcal{L}^{w}(i)$, $\mathcal{C}^{w}(i)$, $\widetilde{\mathcal{M}}(i)$, $\widetilde{\mathcal{L}}^{w}(i)$, and $\widetilde{\mathcal{C}}^{w}(i)$, for $w \in S$, are computed by using information gathered during previous iterations, according to the properties in Lemmas 1 and 2. In more details, the algorithm Algo1 works as follows. It uses an array C indexed by the elements of S, such that, for each word $w \in S$, the entry $\mathsf{C}[w]$ maintains successively the counters $\mathcal{C}^{w}(i)$ and $\widetilde{\mathcal{C}}^{w}(i)$, for $-n \leq i < |\tau|$; more precisely, during iteration i, just before the execution of the conditional test of line 13, we have that $\mathsf{C}[w] = \widetilde{\mathcal{C}}^{w}(i)$, whereas, before the execution of the instruction of line 22 we have that $\mathsf{C}[w] = \mathcal{C}^{w}(i)$. The sets $\widetilde{\mathcal{L}}^{w}(i)$ and $\mathcal{L}^{w}(i)$ are maintained by means of an array Q of $|S|$ queues indexed by the elements of S. Specifically, for each word $w \in S$ and each i, during iteration i of Algo1, just before the conditional test in line 13 (resp., before the instruction of line 22), the queue $\mathsf{Q}[w]$ contains the elements of the set $\widetilde{\mathcal{L}}^{w}(i)$ (resp., $\mathcal{L}^{w}(i)$) increasingly ordered, with the smallest element at the head of the queue and the largest one at the tail. We assume that each queue $\mathsf{Q}[w]$ supports the following operations: (a) `dequeue`$(\mathsf{Q}[w])$, which removes the element at the head of $\mathsf{Q}[w]$; (b) `enqueue`$(\mathsf{Q}[w], j)$, which adds the number j at the tail of the queue $\mathsf{Q}[w]$; (c) `head`$(\mathsf{Q}[w])$, which returns (but not remove) the element at the head of $\mathsf{Q}[w]$; (d) `size`$(\mathsf{Q}[w])$, which returns the number of elements currently contained in the queue $\mathsf{Q}[w]$. The algorithm Algo1 also uses the integer variable M to store successively the values $\mathcal{M}(-n), \mathcal{M}(-n+1), \mathcal{M}(-n+2), \ldots$ as they are

```
Algo1(τ,S,n)
 1.  M := −n − 1
 2.  for w ∈ S do
 3.    Q[w] := ∅
 4.    C[w] := 0
 5.  for i := −n + 1 to |τ| − 1 do
 6.    if τ_{i−1} ∈ S then
 7.      C[τ_{i−1}] := C[τ_{i−1}] − 1
 8.      if size(Q[τ_{i−1}]) > 0 AND
                    (C[τ_{i−1}] − size(Q[τ_{i−1}])) = 0 then
 9.        M := head(Q[τ_{i−1}])
10.        for w ∈ S do
11.          while (size(Q[w]) > 0 AND
                                    head(Q[w]) ≤ M) do
12.            dequeue(Q[w])
13.    if τ_{i+n−1} ∈ S then
14.      C[τ_{i+n−1}] := C[τ_{i+n−1}] + 1
15.      if C[τ_{i+n−1}] = 1 then
16.        M := i + n − 1
17.        for w ∈ S do
18.          while (size(Q[w]) > 0) do
19.            dequeue(Q[w])
20.      else
21.        enqueue(Q[τ_{i+n−1}], i + n − 1)
22.    if M < i − 1 then M := i − 1
23.    output(M) ◁ outputs the value M_{τ,S,n}(i)
```

Fig. 2. The algorithm Algo1 for Problem 2

computed during the execution. Note that at iteration i of Algo1, just after the execution of instruction at line 9, the variable M stores the intermediate value $\widetilde{\mathcal{M}}(i)$, whereas, after the execution of instruction at line 16, M holds the value $\mathcal{M}(i)$.

Note that each queue of the array Q can be efficiently implemented as a circular array of size n in such a way that the supported

operations `enqueue`, `dequeue`, `head`, and `size` take constant execution time each. Using such an implementation, the space complexity of the algorithm Algo1 is readily seen to be $\mathcal{O}(|S| \cdot n)$. Concerning the time complexity of the algorithm Algo1, we must first be explicit on the representation of the input set S. In fact, the particular representation used for S may affect, for instance, the implementation of the conditional tests in lines 6 and 13, and hence their computational costs. We shall assume that the set S is represented as a *map* $\mathbb{S} : S \cup \langle\tau\rangle \longrightarrow [-1, |S| - 1]$ such that

(a) $\mathbb{S}(w) = -1$ if and only if $w \notin S$, for every word $w \in S \cup \langle\tau\rangle$, and
(b) $\mathbb{S}(w') \neq \mathbb{S}(w'')$, for all distinct words $w', w'' \in S \cup \langle\tau\rangle$.

Thus, for each word $w \in S$, the value $\mathbb{S}(w)$ corresponds to the index of w in the array Q. By using this representation, the conditional tests in lines 6 and 13 of Algo1 can just be expressed in the form $\mathbb{S}(\tau_i) \neq -1$ and $\mathbb{S}(\tau_{i+n-1}) \neq -1$, respectively. In the analysis below we shall assume that, for every word $w \in S \cup \langle\tau\rangle$, the value $\mathbb{S}(w)$ can be computed in constant time $\mathcal{O}(1)$,[5] so that the above two tests will take constant execution time. Thus, returning to the running time of Algo1, note that, for each $w \in S$, the **while-loop**s of lines 11 and 18 are executed at most n times

[5] Such a requirement can be achieved as follows. Let $\boldsymbol{W}$ be a fixed set of words (the *vocabulary*), which includes all of the words we are dealing with, so that, in particular, $S \cup \langle\tau\rangle \subseteq \boldsymbol{W}$. We use a *hash function* h which injectively maps each word $w \in \boldsymbol{W}$ to its *hash code* $h(w)$, a nonnegative integer number. Then, we store the map $\mathbb{S}$ in a table T of size $\max\{h(w) : w \in \boldsymbol{W}\}$, in such a way that (a) $T[h(w)] = \mathbb{S}(w)$, for $w \in S$, and (b) $T[h(w)] = -1$, for $w \in \boldsymbol{W} \setminus S$. Thus, the calculation of $\mathbb{S}(w)$ reduces to the extraction of the value $T[h(w)]$, for each word w. Now, the function h can be chosen in such a way that the computation of $h(w)$ can be performed in $\mathcal{O}(|w|)$ time, for each word $w \in \boldsymbol{W}$. Hence, since the maximum length of words in $\boldsymbol{W}$ is fixed (and in fact, in practical cases this quantity is "small enough"), it turns out that the function h can be computed in constant time, yielding a constant execution time for computing $\mathbb{S}(w)$, for each word w. Note also that an additional integer variable could be used to hold the number of entries in T containing values different from -1. This would allow one to retrieve the cardinality of S in constant time.

since each queue $\mathsf{Q}[w]$ has size at most n, and so—under the above assumption concerning the representation of S—the running time of Algo1 is easily seen to be $\mathcal{O}(|\tau| \cdot |S| \cdot n)$.[6] However, this is just a rough estimate. A slightly more accurate analysis leads to an $\mathcal{O}(|\tau| \cdot \max(|S|, n))$ time complexity. Indeed, note that, for each $w \in S$, the queue $\mathsf{Q}[w]$ contains at most all of the occurrences of w in the window $\tau_{[i,i+n-1]}$. Thus, the sum of the sizes of the queues in the array Q, i.e., $\sum_{w \in S} \mathtt{size}(\mathsf{Q}[w])$, is bounded above by n. Hence, the **for-loop**s of lines 10 and 17 are executed $\mathcal{O}(\max(|S|, n))$ times, and therefore the overall time complexity of Algo1 is in fact $\mathcal{O}(|\tau| \cdot \max(|S|, n))$.

The basic algorithm Algo1 presented above can be slightly modified to a $\mathcal{O}(|\tau|)$-time variant, still retaining the same space complexity. Note in fact that there is no need to update simultaneously all of the queues of the array Q during the **for-loop**s of lines 10 and 17; indeed, for each $w \in S$, we can safely postpone the updating of the queue $\mathsf{Q}[w]$ to the next time that w is encountered during the scan of τ, without affecting the correctness of the algorithm. This observation leads to the variant Algo2 of Algo1 reported in Fig. 3. As anticipated, the running time of Algo2 is $\mathcal{O}(|\tau|)$. To see this, consider the behaviour of the queue $\mathsf{Q}[w]$, for an arbitrary word $w \in S$. Let $\mathsf{Occs}(w)$ be the number of occurrences of w in τ. During the execution of Algo2, the number of enqueue operations involving $\mathsf{Q}[w]$ is no larger than the number of positions i such that $\tau_{i+n-1} = w$ and hence no larger than $\mathsf{Occs}(w)$. Thus the number of dequeue operations involving $\mathsf{Q}[w]$ is at most $\mathsf{Occs}(w)$. The total cost of the execution of the two **while-loop**s of lines 7 and 13 is therefore bounded above by the quantity $\sum_{w \in S} \mathsf{Occs}(w)$ which is at most equal to $|\tau|$. Thus, it plainly follows that the running time of Algo2 is in fact $\mathcal{O}(|\tau|)$.

Next we turn to Problem 3. We can solve this problem by means of the following simple iterative procedure. As in the case of algorithms Algo1 and Algo2, a window of length n is iteratively

[6] All algorithms in the rest of the paper will implicitly use for the input set S the same representation by a map $\mathbb{S}$ as above.

```
Algo2(τ,S,n)
 1.  M := −n − 1
 2.  for w ∈ S do
 3.    Q[w] := ∅
 4.    C[w] := 0
 5.  for i := −n + 1 to |τ| − 1 do
 6.    if τ_{i−1} ∈ S then
 7.      while (size(Q[τ_{i−1}]) > 0 AND
                                  head(Q[τ_{i−1}]) ≤ M) do
 8.        dequeue(Q[τ_{i−1}])
 9.      C[τ_{i−1}] := C[τ_{i−1}] − 1
10.      if size(Q[τ_{i−1}]) > 0 AND
                        (C[τ_{i−1}] − size(Q[τ_{i−1}])) = 0 then
11.        M := head(Q[τ_{i−1}])
12.    if τ_{i+n−1} ∈ S then
13.      while (size(Q[τ_{i+n−1}]) > 0 AND
                                  head(Q[τ_{i+n−1}]) ≤ M) do
14.        dequeue(Q[τ_{i+n−1}])
15.      C[τ_{i+n−1}] := C[τ_{i+n−1}] + 1
16.      if C[τ_{i+n−1}] = 1 then
17.        M := i + n − 1
18.      else
19.        enqueue(Q[τ_{i+n−1}], i + n − 1)
20.    if M < i − 1 then M := i − 1
21.    output(M) ◁ outputs the value M_{τ,S,n}(i)
```

Fig. 3. The variant Algo2 of Algo1 for Problem 2

advanced along the text τ, moving it to the right one position at a time, starting from the initial position $i = -n$. We use an array R of length $|S|$, indexed by the elements of S, whose entry $\mathsf{R}[w]$ maintains successively the number of occurrences of the word w in the window starting at the current position i, for each word $w \in S$. An integer variable K is used to count the number of words of S that are contained in the window starting at position i. More for-

mally, for each $i \geq -n$, when the iteration relative to position i is completed, we have:

(a) the entry $\mathsf{R}[w]$ of the array R contains the quantity $\mathcal{C}^w_{\tau,n}(i)$, for each word $w \in S$;
(b) the variable K holds the value $\mathcal{K}_{\tau,S,n}(i)$.

Note that initially, i.e., when $i = -n$, we have that: (i) $\mathsf{R}[w] = 0$, for each word $w \in S$; and (ii) $\mathsf{K} = 0$. Next we describe how the two items R and K are updated when moving from position $i-1$ to position i. Let R_{i-1} and K_{i-1} be the contents of R and K, respectively, just after position $i-1$ has been processed. The following four cases arise.

Case 1) $\tau_{i-1}, \tau_{i+n-1} \notin S$. Plainly, in this case no entry of the array R needs to be updated, nor the variable K needs to be changed.

Case 2) $\tau_{i-1} \in S$ and $\tau_{i+n-1} \notin S$. Since $\tau_{i-1} \in S$ and $\tau_{i-1} \neq \tau_{i+n-1}$, the number of occurrences of τ_{i-1} in the current window $\tau_{[i,i+n-1]}$ is equal to the number $\mathsf{R}_{i-1}[\tau_{i-1}]$ of occurrences of τ_{i-1} in the previous window $\tau_{[i-1,i+n-2]}$ decreased by 1. Thus the entry $\mathsf{R}[\tau_{i-1}]$ is updated to the value $\mathsf{R}_{i-1}[\tau_{i-1}] - 1$. Moreover, if $\mathsf{R}_{i-1}[\tau_{i-1}] = 1$, then the current window $\tau_{[i,i+n-1]}$ does not contain any occurrence of the word τ_{i-1}, whereas the previous window $\tau_{[i-1,i+n-2]}$ contains exactly one occurrence of τ_{i-1}, and therefore the variable K is updated to the value $\mathsf{K}_{i-1} - 1$. When $\mathsf{R}_{i-1}[\tau_{i-1}] \neq 1$, then, as in **Case 1)**, we do nothing.

Case 3) $\tau_{i-1} \notin S$ and $\tau_{i+n-1} \in S$. Since $\tau_{i+n-1} \in S$ and $\tau_{i-1} \neq \tau_{i+n-1}$, the number of occurrences of τ_{i+n-1} in the current window $\tau_{[i,i+n-1]}$ is equal to the number $\mathsf{R}_{i+n-1}[\tau_{i+n-1}]$ of occurrences of τ_{i+n-1} in the previous window $\tau_{[i-1,i+n-2]}$ increased by 1. Thus, the entry $\mathsf{R}[\tau_{i+n-1}]$ is updated to the value $\mathsf{R}_{i-1}[\tau_{i+n-1}]+1$. Moreover, if $\mathsf{R}_{i-1}[\tau_{i+n-1}] = 0$, then the current window $\tau_{[i,i+n-1]}$ contains exactly one occurrence of the word τ_{i+n-1}, whereas the previous window $\tau_{[i-1,i+n-2]}$ does

```
Algo3(τ,S,n)
 1.  K := 0
 2.  for w ∈ S do
 3.     R[w] := 0
 4.  for i := −n + 1 to |τ| − 1 do
 5.     if τ_{i−1} ∈ S then
 6.        R[τ_{i−1}] := R[τ_{i−1}] − 1
 7.        if R[τ_{i−1}] = 0 then K := K − 1
 8.     if τ_{i+n−1} ∈ S then
 9.        R[τ_{i+n−1}] := R[τ_{i+n−1}] + 1
10.        if R[τ_{i+n−1}] = 1 then K := K + 1
11.  output(K) ◁ outputs the value K_{τ,S,n}(i)
```

Fig. 4. The algorithm Algo3 for Problem 3

not contain any occurrence of τ_{i+n-1}, and therefore the variable K is updated to the value $\mathsf{K}_{i-1}+1$. When $\mathsf{R}_{i-1}[\tau_{i+n-1}] \neq 0$, we do nothing, as in **Case 1)**.

Case 4) $\tau_{i-1}, \tau_{i+n-1} \in S$. In this case we simply perform the operations involved in **Case 2)**, followed by the operations involved in **Case 3)**.

The procedure described above translates into the algorithm Algo3 reported in Fig. 4.

Plainly, the space complexity of Algo3 is $\mathcal{O}(|S|)$; moreover, its running time can be readily seen to be $\mathcal{O}(|\tau|)$.

3.1 *Efficiently Solving the Main Problem*

Algorithms Algo2 and Algo3 can be combined into a single iterative algorithm that solves our main Problem 1 in time $\mathcal{O}(|\tau|)$ and space $\mathcal{O}(|A| \cdot n + |B|)$, where we recall that τ is the text, n is the window-length, and A and B are the sets of words to be searched for in τ. The complete pseudo-code of the resulting algorithm, named AlgoMain, is reported in Fig. 5–6.

```
AlgoMain(τ,A,B,n)
 1.  M := −n − 1
 2.  for w ∈ A do
 3.     Q[w] := ∅
 4.     C[w] := 0
 5.  A := 0
 6.  K := 0
 7.  for w ∈ B do
 8.     R[w] := 0
 9.  E := −2n
10.  Ω+ := 0
11.  Ω− := 0
12.  for i := −n + 1 to |τ| do
13.     if τ_{i−1} ∈ A then
14.        while (size(Q[τ_{i−1}]) > 0 AND
                                  head(Q[τ_{i−1}]) ≤ M) do
15.           dequeue(Q[τ_{i−1}])
16.        C[τ_{i−1}] := C[τ_{i−1}] − 1
17.        if (C[τ_{i−1}] − size(Q[τ_{i−1}]) = 0) AND
                                          A = |A| then
18.           if (M − E < n) then Ω+ := Ω+ + 1
19.           else Ω− := Ω− + 1
20.        if C[τ_{i−1}] = 0 then A := A − 1
21.        if size(Q[τ_{i−1}]) > 0 AND
                        (C[τ_{i−1}] − size(Q[τ_{i−1}])) = 0 then
22.           M := head(Q[τ_{i−1}])
```

Fig. 5. The algorithm AlgoMain for Problem 1, part 1

Note that the block of instructions from line 1 to line 4 (resp., from line 6 to line 8) of algorithm AlgoMain corresponds to the initialization of algorithm Algo2 (resp., algorithm Algo3), whereas the instructions from line 13 to line 32, excluding lines 17, 18, 19, 20 and 28, correspond to the body of the main **for-loop** of algorithm Algo2. Also, observe that the instructions from line 33 to

```
23.     if τ_{i+n-1} ∈ A then
24.       while (size(Q[τ_{i+n-1}]) > 0 AND
                                  head(Q[τ_{i+n-1}]) ≤ M) do
25.         dequeue(Q[τ_{i+n-1}])
26.       C[τ_{i+n-1}] := C[τ_{i+n-1}] + 1
27.       if C[τ_{i+n-1}] = 1 then
28.         A := A + 1
29.         M := i + n - 1
30.       else
31.         enqueue(Q[τ_{i+n-1}], i + n - 1)
32.     if M < i - 1 then M := i - 1
33.     if τ_{i-1} ∈ B then
34.       R[τ_{i-1}] := R[τ_{i-1}] - 1
35.       if R[τ_{i-1}] = 0 then K := K - 1
36.     if τ_{i+n-1} ∈ B then
37.       R[τ_{i+n-1}] := R[τ_{i+n-1}] + 1
38.       if R[τ_{i+n-1}] = 1 then K := K + 1
39.     if K = |B| then E := i
40.     else
41.       if (E < n - i) then E := n - i
42.   output(Ω^+/(Ω^+ + Ω^-)) ◁ outputs the closeness factor
                                                of A to B
```

Fig. 6. The algorithm AlgoMain for Problem 1, part 2

line 38 of algorithm AlgoMain correspond to the body of the main **for-loop** of algorithm Algo3. The additional integer variables A and E are used to maintain successively the values $\mathcal{K}_{\tau,A,n}(i)$ and $\mathcal{E}_{\tau,B,n}(i)$, for $i \geq -n$, as they are computed during the execution of the algorithm; thus, at the end of iteration i of AlgoMain we have $\mathsf{A} = \mathcal{K}_{\tau,A,n}(i)$ and $\mathsf{E} = \mathcal{E}_{\tau,B,n}(i)$. Note that the variable E is successively updated according to relation (1) in Section 3. Finally, the two variables Ω^+ and Ω^- are used to hold the values $\Omega^+_{\tau,A,n}(B)$ and $\Omega^-_{\tau,A,n}(B)$; more precisely, at the end of the exe-

cution of the main **for-loop** of line 12 of AlgoMain, we have that $\Omega^+ = \Omega^+_{\tau,A,n}(B)$ and $\Omega^- = \Omega^-_{\tau,A,n}(B)$. In particular, concerning the calculation of the values $\Omega^+_{\tau,A,n}(B)$ and $\Omega^-_{\tau,A,n}(B)$, AlgoMain determines whether $\tau_{[i-1,\mathcal{M}_{\tau,A,n}(i-1)]}$ is a minimal window containing A, for $i = 1, 2, \ldots, |\tau|$, in the following way. At iteration i, initially AlgoMain checks whether $\tau_{i-1} \in A$ (see the instruction at line 13). If $\tau_{i-1} \notin A$, then, plainly, the window $\tau_{[i-1,\mathcal{M}_{\tau,A,n}(i-1)]}$ cannot be a minimal window containing A, and in fact, in this case, no update of the variables Ω^+ and Ω^- takes place. On the other hand, if $\tau_{i-1} \in A$ holds, AlgoMain successively updates the queue $\mathsf{Q}[\tau_{i-1}]$ and the entry $\mathsf{C}[\tau_{i-1}]$ by executing the **while-loop** at lines 14–15 and the instruction at line 16, respectively. In fact, immediately after the execution of the instruction at line 16, we have that $\mathsf{C}[\tau_{i-1}] - \texttt{size}(\mathsf{Q}[\tau_{i-1}])$ is equal to the number of occurrences of the word τ_{i-1} in the window $\tau_{[i,\mathcal{M}_{\tau,A,n}(i-1)]}$, as can be readily verified. In addition, the variable A contains the number of words in A that occur in the window $\tau_{[i-1,i+n-2]}$. Therefore, $\tau_{[i-1,\mathcal{M}_{\tau,A,n}(i-1)]}$ is a minimal window containing A iff $\mathsf{C}[\tau_{i-1}] - \texttt{size}(\mathsf{Q}[\tau_{i-1}])$ is equal to 0 and A is equal to $|A|$, which corresponds in fact to the conditional test at line 17.

By inspection, it is not difficult to verify that AlgoMain has an $\mathcal{O}(|A| \cdot n + |B|)$ space complexity; moreover, by using considerations similar to those made for algorithms Algo2 and Algo3, and assuming that the quantities $|A|$ and $|B|$ can be computed in constant time,[7] the running time of AlgoMain can be readily seen to be $\mathcal{O}(|\tau|)$.

We conclude the section with a simple example illustrating the idea, sketched in Section 1, for simplifying a text by means of the closeness factor of word sets.

Example 1. Let the words we are dealing with be represented by the following symbols:

A B C D E F

[7] See footnote 5.

and let

$$\tau = \text{DCCDAADEFFEABBCFD} \quad (2)$$
$$\text{EEFABCDCBDDCEAAEBCBC}$$

be the corpus and $\sigma = \text{DACBDCB}$ be the text we intend to simplify. Moreover, let us assume that the maximum word distance n is 6 and that the threshold t is 0.7. Let $T = \langle\sigma\rangle = \{\text{A}, \text{B}, \text{C}, \text{D}\}$ be the set of the words occurring in σ. We start by selecting the first candidate word $w_1 = \text{A}$ and try to remove it from T. This involves computing the closeness factor $(T \setminus \{w_1\} \ltimes T)_{\tau,n}$ and then comparing it with the threshold t. By executing the algorithm AlgoMain, we get $(T \setminus \{w_1\} \ltimes T)_{\tau,n} = 4/5$.[8] Since $4/5 > t$ we can safely remove the word w_1 from T thus obtaining the word set $T_1 = T \setminus \{w_1\} = \{\text{B}, \text{C}, \text{D}\}$. Subsequently, we select the second candidate word $w_2 = \text{C}$ to be removed from T_1 and compute the closeness factor $(T_1 \setminus \{w_2\} \ltimes T_1)_{\tau,n}$, obtaining $(T_1 \setminus \{w_2\} \ltimes T_1)_{\tau,n} = 4/5$.[9] As before, since this value is greater than the threshold t, we remove w_2 from T_1 and form the word set $T_2 = T_1 \setminus \{w_2\} = \{\text{B}, \text{D}\}$. At this point, it can be verified that, for every word $w \in T_2$, the closeness factor $(T_2 \setminus \{w\} \ltimes T_2)_{\tau,n}$ is less than t,[10] so that no further word can be removed and the process terminates. Hence, the words that have been removed are $w_1 = \text{A}$ and $w_2 = \text{C}$ which we then eliminate from σ, obtaining the simplified text DBDB. Note that, if at the second step we had selected the word $w_2 = \text{B}$, rather than $w_2 = \text{C}$, we would have obtained $(T_1 \setminus \{w_2\} \ltimes T_1)_{\tau,n} = 5/7$ which is less than the threshold t, and so we could not have removed this word from T_1. Similarly, the

[8] In fact, in the corpus τ there are exactly five minimal windows containing $T \setminus \{w_1\} = \{\text{B}, \text{C}, \text{D}\}$, namely the windows starting at positions 13, 21, 23, 24 and 25; of these windows, only the one starting at position 24 cannot be extended to a window containing T. Thus $\Omega^{+}_{\tau,T\setminus\{w_1\},n}(T) = 4$ and $\Omega^{-}_{\tau,T\setminus\{w_1\},n}(T) = 1$ which yield $(T \setminus \{w_1\} \ltimes T)_{\tau,n} = 4/5$.

[9] In fact, it can be verified that $\Omega^{+}_{\tau,T_1\setminus\{w_2\},n}(T_1) = 4$ and $\Omega^{-}_{\tau,T_1\setminus\{w_2\},n}(T_1) = 1$.

[10] More precisely, we have that $(\{\text{B}\} \ltimes \{\text{B}, \text{D}\})_{\tau,n} = (\{\text{D}\} \ltimes \{\text{B}, \text{D}\})_{\tau,n} = 4/6$.

choice $w_2 = \mathrm{D}$ would have not led to any removal as well, since, in this case, we would have had $(T_1 \setminus \{w_2\} \ltimes T_1)_{\tau,n} = 3/7 < t$, as can easily be verified.

4 CONCLUSIONS

We have introduced a statistical, corpus-based measure of the closeness of two sets A and B of words in texts—the *closeness factor*—and we have presented an efficient algorithm for computing it. Our proposed algorithm involves a window-based search of the words of the sets A and B through a corpus τ, under the restriction that the words of the sets should co-occur within a given maximum distance n.

The algorithm runs in $\mathcal{O}(|\tau|)$-time, where $|\tau|$ is the length of τ, and uses $\mathcal{O}(|A| \cdot n + |B|)$-space, where $|A|$ and $|B|$ are the cardinalities of A and B, respectively. The closeness factor has applications in various subfields of natural language processing, among which text simplification and sentiment analysis.

We plan to investigate further problems involving the closeness factor, such as the optimization problem mentioned in Section 1.

REFERENCES

1. Indurkhya, N., Damerau, F.J. (eds.) 2010. *Handbook of Natural Language Processing*. 2nd ed. CRC, Boca Raton, FL.
2. Radinsky, K., Agichtein, E., Gabrilovich, E., Markovitch, S. 2011. A word at a time: Computing word relatedness using temporal semantic analysis. In proceedings of the *20th International Conference on World Wide Web* (pp. 337-346), 11, New York, NY, USA, ACM.
3. Reisinger, J., Mooney, R. J. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 109-117).
4. Lowe, W. 2001. Towards a theory of semantic space. In J. T. Moore, & K. Stenning (Eds.), *Proceedings of the 23rd Conference of the Cognitive Science Society* (pp. 576-581).

5. Lee, L. 1999. Measures of distributional similarity. In proceedings of the *37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, Morristown, NJ, USA, Association for Computational Linguistics (pp. 25-32).
6. Fellbaum, C. (ed) 1998. *WordNet: An Electronic Lexical Database*. MIT Press.
7. Budanitsky, A. & Hirst, G. 2006. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32, 13-47.
8. Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Pasca, M., Soroa, A. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In proceedings of *NAACL-HLT 09*.
9. Borz`ı, V., Faro, S. & Pavone, A. 2014. Automatic extraction of semantic relations by using web statistical information. In N. Hernandez, R. J¨aschke & M. Croitoru (Eds.), *Graph-Based Representation and Reasoning*. *Lecture Notes in Computer Science*. Springer International Publishing (pp. 174-187).
10. Siddharthan, A. 2006. Syntactic simplification and text cohesion. *Research on Language and Computation*, 4/1, 77-109
11. Pang, B. & Lee, L. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2/1-2, 1-135.

ACKNOWLEDGMENTS

We would like to thank the anonymous referee for his/her helpful comments. This work has been partially supported by the project PRISMA PON04a2 A (funded by the Italian Ministry of University and Research within the PON 2007-2013 "Smart cities and communities" framework) and by the FIR-2014 project "COMPACT: COMPuting reliably on AuthentiCated Texts" (funded by the University of Catania).

DOMENICO CANTONE
DIPARTIMENTO DI MATEMATICA E INFORMATICA
UNIVERSITA DI CATANIA, VIALE ANDREA DORIA N. 6,
I-95125 CATANIA, ITALY.

SALVATORE CRISTOFARO
DIPARTIMENTO DI MATEMATICA E INFORMATICA
UNIVERSITA DI CATANIA, VIALE ANDREA DORIA N. 6,
I-95125 CATANIA, ITALY.

GIUSEPPE PAPPALARDO
DIPARTIMENTO DI MATEMATICA E INFORMATICA
UNIVERSITA DI CATANIA, VIALE ANDREA DORIA N. 6,
I-95125 CATANIA, ITALY.

International Journal of Computational Linguistics and Applications, Vol. 6, No. 1, 2015, pp. 173–188
Received 25/02/2015, Accepted 20/05/2015, Final 12/06/2015.
ISSN 0976-0962, http://ijcla.bahripublications.com

Bag-of-Concepts Document Representation for Textual News Classification

MARCOS MOURIÑO-GARCÍA
ROBERTO PÉREZ-RODRÍGUEZ
LUIS ANIDO-RIFÓN
University of Vigo, Vigo, Spain

ABSTRACT

Automatic classification of news articles is a relevant problem due to the large amount of news generated every day, so it is crucial that these news are classified to allow for users to access to information of interest quickly and effectively.

Traditional classification systems represent documents as bag-of-words (BoW), which are oblivious to two problems of language: synonymy and polysemy. This paper shows the advantages of using a bag-of-concepts (BoC) representation of documents, which tackles synonymy and polysemy, in text news classification – using a Support Vector Machines algorithm. In order to create BoC representations, a Wikipedia-based semantic annotator is used.

To evaluate the proposal we used a purpose-built corpus and the Reuters - 21578 corpus. Results show that the efficiency of the BoC approach is very dependent on the performance of the semantic annotator in extracting concepts, which depends heavily on the characteristics of particular corpora, reaching performance increases up to 29.65%.

1. INTRODUCTION

The information and communication society entails the existence of huge amounts of information distributed across and along the Internet. That information is being continuously created by a lot of sources. Besides, the demand of information by users is

growing day by day, which makes necessary and essential to automate the ordering of information. The automatic classification of text documents into a predefined set of categories is a field that has a large number of applications and provides a solution to the problem presented above. Among these applications, we can include: the classification of books by theme, genre, or subject; the classification of online educational resources into their subject area or educational level; the classification of blogs by their topic; and the classification of textual news in its proper category. Referring again to the amount of available information on the Internet, there are a lot of sources that generate immense amounts of daily news. It is, therefore, necessary that those news can be organized or categorised into a finite set of categories, in such a way that it allows an easy, quick, and efficient access to those that are of interest – i.e. it is crucial that these news are classified.

Automatic text classification uses supervised machine learning techniques. First, the classification algorithm is selected – there are many classification algorithms, being the most relevant in the state of the art k-Nearest Neighbour, Decision Tree, Neural Networks, Bayes and Support Vector Machines [1]. Next, the training sequence is selected – a set of examples whose category is known, which serves to train the classifier. Finally, the algorithm receives a test sequence – a set of documents whose category is unknown – so that it may predict the most appropriate category where to classify each document, making use of what was learnt in the training phase.

Natural Language Processing (NLP) techniques represent documents based on features contained in them, such as the structure of the document itself, the words that it comprises, or the frequency of these in the text [2]. Automatic classification of documents makes use of these techniques, so that a classifier can predict to which category a given document belongs to simply on the basis of some features of the aforesaid. Although there are numerous representations, the most commonly used is VSM (Vector Space Model) [3], in which each document belonging to a collection is represented as a point in space, commonly using as

weights the frequency of occurrence of words. This representation is known as bag-of-words, begin a bag – or multiset – a set of elements that can occur several times [4]. Thus, using this model, a document is represented by a set of words and the frequency of occurrence of these in the text. This model does not tackle two common problems language: synonymy and polysemy [5,6,7,8,9,10,11]. The problem of synonymy means that synonyms are not unified, whereas the problem of polysemy means that a word can have several meanings. For example, when a classifier is trained with a set of examples that contains the word "car", which belongs to "motor" category, in the moment of classifying a new document that contains the word "automobile" previous training will not be useful because words "car" and "automobile" are different – problem of synonymy. Regarding polysemy, when a document that contains the word "mercury" is classified into "astronomy" category, it may cause errors when classifying another document that contains the word "mercury", this time making reference to a different meaning of "mercury" like the chemical element or the Roman god.

In order to solve the problems introduced by synonymy and polysemy, some authors have proposed a concept-based document representation, defining the concept as "unit of meaning" [11,12,13]. Following this model, documents are represented by a weighted bag-of-concepts. By definition, the concepts are not ambiguous, so that they eliminate the problems introduced by synonymy and polysemy, providing promising results in text classification tasks [14].

In the literature, there are several proposals for creating BoC document representations, and different ways to represent a concept. Deerwester et al. [15] and Landauer et al. [16] propose Latent Semantic Analysis (LSA), where a concept is defined as a vector that represents the occurrence of a term in certain contexts. The main advantage of this approach is that it deals with synonymy, but it does not combat polysemy. The second approach is Explicit Semantic Analysis (ESA), proposed by Gabrilovich and Markovitch [17], where a concept is an entry in a external database used as background knowledge – Wikipedia,

Wordnet, Open Directory Project, etc. Thus, each document is annotated in accordance with its overlap with each entry in the knowledge base. The main problem with this approach is the generation of outliers [5], being an outlier a concept that is not related to the document to annotate. The third proposal for creating BoC representations – the one that we use in this paper – is based on semantic annotators. A semantic annotator is a software agent that is responsible for extracting the concepts that define a document, linking these concepts with entries from external sources such as Wikipedia. Semantic annotators also perform word sense disambiguation – thus tackling synonymy and polysemy – and they assign a weight to each extracted concept in accordance with their relevance in the text.

We consider that there is a research gap on the use of a Support Vector Machines (SVM) classifier with a BoC representation of documents, as well as on their application to create a classifier of textual news into a finite set of categories. This paper aims at providing solutions to this problem by designing, developing, and empirically evaluating an automatic system that classifies online text news using machine learning techniques and that follows the bag-of-concepts paradigm. The evaluation of the system was performed by conducting several empirical experiments with two corpora: Reuters-21578 and a purpose-built corpus that comprises news of the Reuters agency, hereinafter call Reuters-27000.

The rest of the paper is organised as follows: the next section conducts a review of the state of the art; Section 3 presents the corpora used, the algorithm and evaluation metrics employed and the proposal; Section 4 shows the results obtained and its analysis; and finally Section 5 presents the conclusions and proposals for future work.

2. LITERATURE REVIEW

To the best of our knowledge, there is not any work about classification of online news using a BoC document representation. This section shows some examples of previous proposals for applying BoC representations to text document

classification. Täckström [8] uses LSA for text categorisation tasks, obtaining positive results using BoC in categories where BoW fails. Yu et al. [18] also obtain good results in classification tasks using LSA and Neural Networks. Gupta and Ratinov [19] report good results using ESA in the classification of small pieces of text, outperforming BoW representations. As for semantic annotators, Huang et al. [6] make use of WLM (Wikipedia Link Measure) – proposed by Milne and Witten [7] – to create a BoC document representation for automatic text classification tasks, using the k-Nearest Neighbour algorithm and the 20Newsgroups corpus for evaluation. Torunoglu et al. [20] use Wiki Concept Extractor to extract the titles of the training sequence documents and use the extracted titles, categories, and redirects to enrich tweets with these topic signatures; the resulting enriched tweets have much more than the original 140 characters.

Among previous works about categorisation of online news are the following ones. Lim et al. [21] propose a classification system that provides good results in classification tasks through the use of the SVM algorithm. Selamat et al. [22] present an approach for online news classification using Neural Networks and providing, according to the authors, acceptable levels or accuracy in datasets composed of sports news. Zhang et al. [23] present a framework for classification of online news using the SVM algorithm and combining different representations and subsets of features as BoW, noun phrases, and named entities; the results presented show that the combination of these subsets of features and representations improve the performance of the classifier when use only BoW representation. Kumar et al. [24] propose in their work a financial news classifier in "rise" and "drop" categories exploiting textual rich contained in the news themselves.

3. RESEARCH METHOD

3.1. *Dataset*

Reuters-27000. Reuters-27000 is a corpus that we expressly created for the evaluation of the proposal presented in this

paper.[1] It comprises about 27,000 online news from Reuters agency, belonging to only one category. After removing duplicates, the corpus results in a set of 23,166 news that belong to one of the 8 following categories: Health, Art, Politics, Sports, Science, Technology, Economy and Business. Besides, this corpus is divided in a training sequence that comprises 10,433 documents and a test sequence composed of 12,733 documents.

Reuters-21578 Reuters-21578 is a corpus that comprises 21,578 Reuters news classified into one or more of 60 categories available. After removing from the corpus those elements belonging to more than one category, the resulting corpus comprises 9,494 documents, divided in a training sequence of 7,595 documents and a test sequence that comprises 1,899 documents.

3.2. *Support Vector Machines (SVM)*

SVM is a set of supervised machine learning algorithms for performing – among other tasks – regression, clustering, and classification. SVM is one of most relevant algorithms found in the state-of-the-art, along with Naïve Bayes, Decision Trees, or k-Nearest Neighbour, among others. The basic idea consists in, given a set of elements each one belonging to one category, SVM algorithms build a model that can predict whether a new element belongs to one category or another. More formally, an SVM is a model that represents the elements as points in space, separating the categories as much as possible. When new items appear in the model, they will be classified into one or another category depending on their proximity to each. [25] provide a more technical and detailed definition.

3.3. *Evaluation metrics*

Sebastiani [26] and Sahlgren and Cöster [14] define the following metrics for the evaluation of automatic text classification:

[1] The corpus is available at http://www.itec-sde.net/reuters_27000.zip

$$P = Precision = \frac{TP}{(TP + FP)} \tag{1}$$

$$R = Recall = \frac{TP}{(TP + FN)} \tag{2}$$

Being TP, TN, FP, FN true positive, true negative, false positive and false negative respectively. Positive means that the document was classified in the category to which it belongs; negative means the opposite; true means that the classification was done correctly; and false means that the classification was done incorrectly.

Furthermore, a measure that combines Precision and Recall, F1-score, is defined. This measure is used to harmonise the two previous measures in order to provide a measure of the performance of the classifier.

$$F_1 = \frac{2 * P * R}{P + R} \tag{3}$$

3.4. *Approach*

The proposal presented in this paper consists in the classification of the two corpora presented in Section 3.1 through the use of an adaptation of the SVM algorithm in order to be trained and tested with documents represented as bags-of-concepts.

First, it is necessary to obtain the BoC representation of documents. To this end, it is necessary to annotate the documents – in other words, to create bags-of-concepts for all of them. As already mentioned, in order to create the BoC representation, we have opted to use semantic annotators, in particular the algorithm proposed by Milne and Witten [27]. This algorithm uses NLP techniques, machine learning, and data mining in Wikipedia. The functioning of the algorithm is based on three steps.

- First step is *candidate selection*. Given a text document that comprises a set of n-grams – being an n-gram a continuous

sequence of n words – the algorithm queries a vocabulary that contains all the anchor texts of Wikipedia to check if any of the n-grams are present in the vocabulary. Thus, the more relevant candidates (n-grams) are those that are used most often as anchor texts in Wikipedia.

- The next step is *disambiguation*. Given the vocabulary of anchor texts, the algorithm selects the most probable target for each of the candidates. This process is based on machine learning, using as training sequence Wikipedia articles, which contain good examples of disambiguation done manually. Disambiguation is performed based on two factors: the relationship with other unambiguous terms of the context, and how common is the relationship between an anchor text and the target Wikipedia article.
- The third and final step is *link detection*, which consists in measuring the relevance of each of the concepts extracted from the text. To this end, machine learning techniques are used again, using as training sequence Wikipedia articles, since each of them is an example of what constitutes a relevant link and what does not. Figure 1 shows graphically the process of obtaining the BoC representation of a text document, being each concept an Wikipedia article.

Once we have obtained the BoC representation of each document, to carry out the proposal we used the *Scikit-learn* library: it is a module for Python, a suite that comprises the main machine learning algorithms in the state-of-the-art [28]. Particularly, we chose the SVM algorithm – defined in Section 3.2 – which corresponds to the *svm.svc.LinearSVC* class in the *Scikit-learn* library.

4. RESULTS AND ANALYSIS

In this section, we present the experiments conducted, the results obtained, and their analysis. The experiments conducted consist in the classification of each corpus described in Section 3.1 using the SVM algorithm and a concept-based document

representation. For the sake of temporal and computational efficiency, to obtain preliminary results that allow us to get an idea of the performance of the proposed system, the experiments have been performed on subsets of the corpora. In the one hand, in the Reuters-21578 corpus, we selected the first 150 training documents for each category as the maximum training sequence, and all the test documents available as the test sequence. In the other hand, in the Reuters-27000 corpus, we also selected as maximum training sequence the first 150 training elements per category, and the first 200 elements of test from each category as the test sequence (1,600 documents).

Figure 2 and Table 1 show the evolution of the F1-score for BoW and BoC varying the length of the training sequence in the Reuters-27000 corpus. We can observe that the BoC representation outperforms the classical BoW, achieving increases up to 29.65%. Thus, the advantages of using BoC are evident, because BoC remove the problems introduced by synonymy and polysemy, increasing the performance of the classifier. We can also note that, as the training sequence increases the graphs converge, because the large amount of data masks the problems introduced by synonymy and polysemy.

Figure 3 and Table 1 show the evolution of the F1-score for BoW and BoC varying the length of the training sequence in Reuters-21578 corpus. In this case, the performance of the BoW representation is clearly superior to BoC. These results clearly show that the performance of BoC representation depends heavily on the ability of the semantic annotator to extract concepts from documents. News in the Reuters-21578 corpus contain lots of abbreviations, measures, and other words that the semantic annotator fails to translate into concepts. In all those cases, the BoW representation performs much better than BoC.

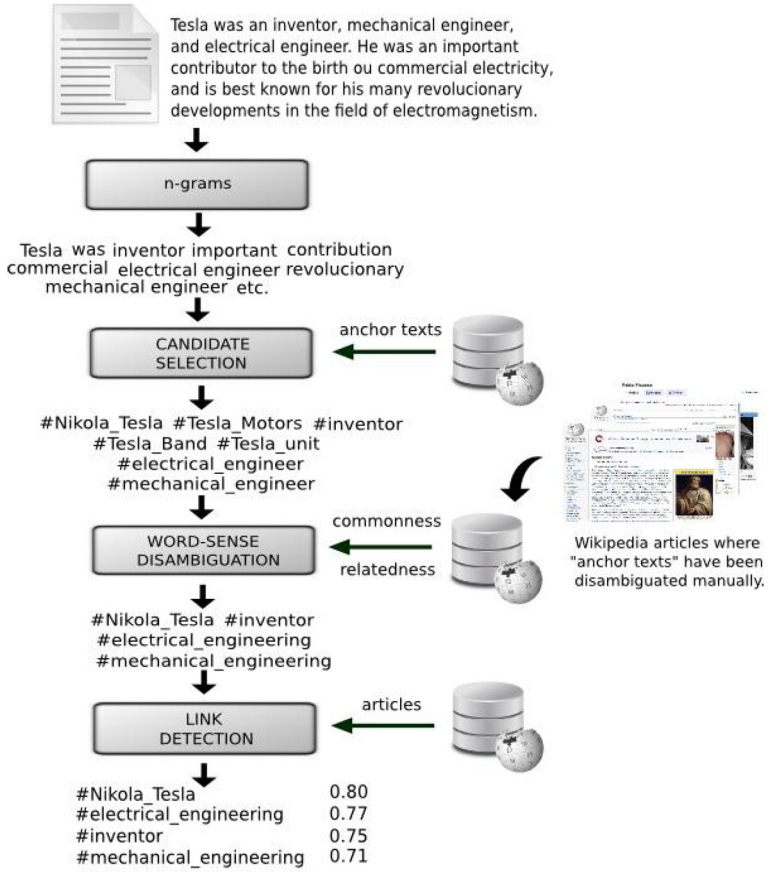


Figure 1. *Automatic extraction of concepts through Milne and Witten [27] algorithm*

Table 2 shows an example of documents of both corpora and the concept that the semantic annotator extracts from them. We can see clearly that the number of concepts extracted from Reuters-27000 document is greater than the number of concepts extracted from Reuters-21578 document. Besides, the quality of concepts extracted from Reuters-27000 document is clearly superior than the quality of concepts extracted from Reuters-21578 document.

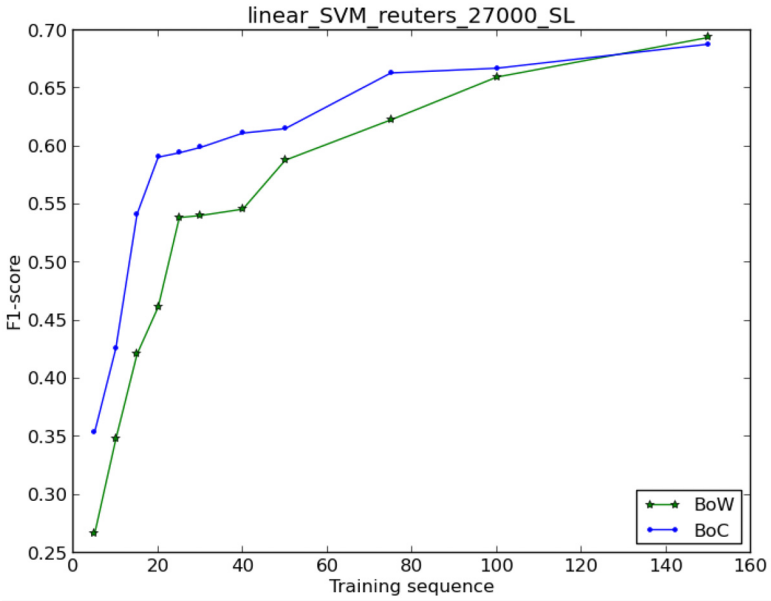


Figure 2. *F1-score for BoW and BoC varying the length of the training sequence in Reuters-27000 corpus*

5. CONCLUSIONS

The study presented in this paper attempts to provide solutions aimed at increasing the performance of automatic news classification systems. To that end, we present an automatic online news classification system – using machine learning techniques and the SVM algorithm – using a BoC representation of the documents that allows for dealing with synonymy and polysemy.

Results obtained show that the performance of the BoC representation depends largely on the ability of the semantic annotator to extract concepts from documents. Thus, we can see that in Reuters-27000, which comprises extensive and well-formed news, the performance offered by BoC outperforms clearly BoW, achieving increases up to 29.65%. In the other hand, documents from Reuters-21578 corpus contains lots of

abbreviations, measures, and other words that the annotator cannot translate into concepts, thus affecting negatively its performance.

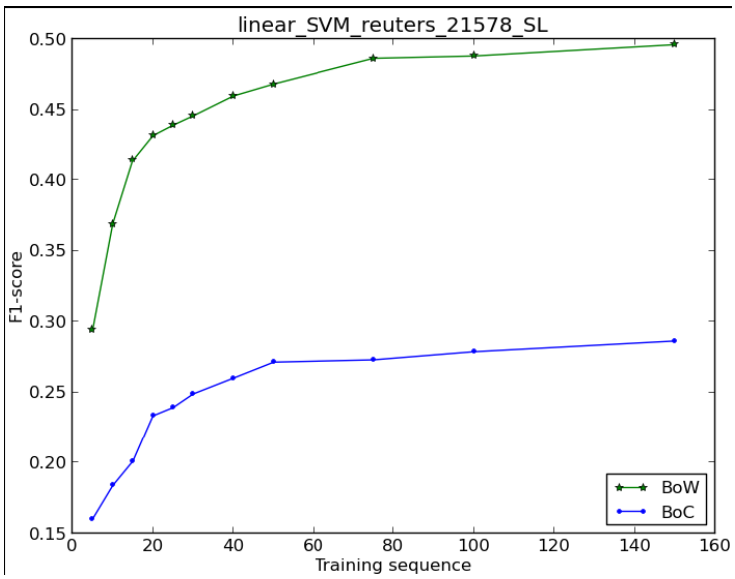


Figure 3. *F1-score for BoW and BoC varying the length of the training sequence in Reuters-21578 corpus*

Table 1. *F1-score for BoW and BoC varying the length of the training sequence in Reuters-27000 and Reuters-21578 corpora*

		5	10	15	20	25	30	40	50	75	100	150
Reuters-27000	BoW	0.267	0.348	0.421	0.462	0.539	0.541	0.546	0.588	0.623	0.660	0.694
	BoC	0.354	0.426	0.542	0.599	0.612	0.591	0.595	0.615	0.663	0.667	0.688
Reuters-21578	BoW	0.294	0.369	0.414	0.432	0.446	0.460	0.439	0.468	0.486	0.488	0.496
	BoC	0.160	0.184	0.201	0.234	0.239	0.249	0.260	0.272	0.279	0.273	0.286

Finally, this study can be extended by: conducting more experiments with other corpora; using different classification algorithms; using other semantic annotators; and even using a hybrid representation of documents, which would possibly take advantage of the benefits of both BoW and BoC representations.

Table 2. *Reuters-21578 and Reuters-27000 documents and concepts extracted from them*

Corpus	**Reuters-21578**	**Reuters-27000**
Category	Earn	Health
Text	Shr 39 cts vs 50 cts Net 1,545,160 vs 2,188,933 Revs 25.2 mln vs 19.5 mln Year Shr 1.53 dlrs vs 1.21 dlrs Net 6,635,318 vs 5,050,044 Revs 92.2 mln vs 77.4 mln NOTE: Results include adjustment of 848,600 dlrs or 20 cts shr for 1986 year and both 1985 periods from improvement in results of its universal life business than first estimated. Reuter	The drug, when given in addition to standard treatment, extended median overall survival in 50 percent of newly-diagnosed glioblastoma multiforme (GBM) patients to two years in a mid-stage study. Usually GBM patients succumb to the disease in one year. (Reporting by Natalie Grover in Bangalore; Editing by Joyjeet Das)
Concepts	Nordisk Mobiltelefon (Sweden) 1986 1985 635 848 933 Universal life insurance Pandan Bikol language Business Universality (philosophy)	Therapy Disease Median Drug Gliobastoma multiforme Bangalore Theatre Editing Standardization Grover Bar (unit) Natalie Cole Standard treatment Survival rate Report

REFERENCES

1. Yang, Y. 1999. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1/1, 69-90.
2. Settles, B. 2010. Active learning literature survey. *Machine Learning*, 15/2, 201-221.

3. Salton, G., Wong, A. & Yang, C. S. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18/11, 613-620.
4. Blizard, W. D. 1988. Multiset theory. *Notre Dame Journal of Formal Logic*, 30/1, 36-66.
5. Egozi, O., Markovitch, S. & Gabrilovich, E. 2011. Concept-based information retrieval using explicit semantic analysis. *ACM Transactions on Information Systems*, 29/2, 1-38.
6. Huang, L., Milne, D., Frank, E. & Witten, I. H. 2012. Learning a concept-based document similarity measure. *Journal of the American Society for Information Science and Technology*, 63/8, 1593-1608.
7. Milne, D. & Witten, I. H. 2008. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. *Proceedings of AAAI Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy* (pp. 25-30).
8. Täckström, O. 2005. *An Evaluation of Bag-of-Concepts Representations in Automatic Text Classification*.
9. Tsao, Y., Chen, K. Y. & Wang, H. M. 2013. Semantic naïve Bayes classifier for document classification. *International Joint Conference on Natural Language Processing* (pp. 1117-1123).
10. Wang, P., Hu, J., Zeng, H.-J., Chen, L. & Chen, Z. 2007. Improving text classification by using encyclopedia knowledge. *Seventh IEEE International Conference on Data Mining (ICDM 2007)* (pp. 332-341), Oct.
11. Wang, P. Hu, J., Zeng, H.-J., Chen, L. & Chen, Z. 2008. Using Wikipedia knowledge to improve text classification. *Knowledge and Information Systems*, 19, 265-281, Sep.
12. Medelyan, O., Witten, I. H. & Milne, D. 2008. Topic indexing with Wikipedia. *Proceedings of the AAAI WikiAI Workshop*, (pp. 19-24).
13. Stock, W. G. 2010. Concepts and semantic relations in information science. *Journal of the American Society for Information Science and Technology*, 61/10, 1951-1969.
14. Sahlgren, M. & Cöster, R. 2004. Using bag-of-concepts to improve the performance of support vector machines in text categorization. *Proceedings of the 20th International Conference on Computational Linguistics*.
15. Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K. & Harshman, R. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41, 391-407.

16. Landauer, T. K. & Dumais, S. T. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104/2, 211-240.
17. Gabrilovich, E. & Markovitch, S. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In proceedings of the *20th International Joint Conference on Artificial Intelligence* (pp. 1606-1611).
18. Yu, B., Xu, Z.-b. & Li, C.-h. 2008. Latent semantic analysis for text categorization using neural network. *Knowledge-Based Systems*, 21/8, 900-904.
19. Gupta, R. & Ratinov, L. 2008. Text categorization with knowledge transfer from heterogeneous data sources. In proceedings of the *23rd National Conference on Artificial Intelligence* (pp. 842-847).
20. Torunoglu, D., Telseren, G., Sagturk, O. & Ganiz, M. C. 2013. Wikipedia based semantic smoothing for twitter sentiment classification, *2013 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA)* (pp. 1-5).
21. Lim, C.-H. C. A. S. E.-P. 2001. Automated online news classification with personalization. In *4th International Conference on Asian Digital Libraries*.
22. Selamat, A., Yanagimoto, H. & Omatu, S. 2002. Web news classification using neural networks based on PCA. In proceedings of the *41st SICE Annual Conference* SICE 2002, Vol. 4.
23. Zhang, Y., Dang, Y., Chen, H., Thurmond, M. & Larson, C. 2009. Automatic online news monitoring and classification for syndromic surveillance. *Decision Support Systems*, 47, 508-517.
24. Kumar, R., Kumar, B. & Prasad, C. 2012. *Financial News Classification using SVM*, 2, 1-6.
25. Hearst, M., Dumais, S., Osman, E., Platt, J. & B. Scholkopf. 1998. *Support Vector Machines*.
26. Sebastiani, F. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34, 1-47.
27. Milne, D. & Witten, I. H. 2013. An open-source toolkit for mining Wikipedia. *Artificial Intelligence*, 194, 222-239.
28. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. 2012. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12, 2825-2830.

MARCOS MOURIÑO-GARCÍA
DEPARTMENT OF TELEMATICS ENGINEERING,
UNIVERSITY OF VIGO, VIGO, SPAIN.

ROBERTO PÉREZ-RODRÍGUEZ
DEPARTMENT OF TELEMATICS ENGINEERING,
UNIVERSITY OF VIGO, VIGO, SPAIN.

LUIS ANIDO-RIFÓN
DEPARTMENT OF TELEMATICS ENGINEERING,
UNIVERSITY OF VIGO, VIGO, SPAIN.