Towards Explainable Text Classification by Jointly Learning Lexicon and Modifier Terms

Jérémie Clos and **Nirmalie Wiratunga** and **Stewart Massie**
Robert Gordon University, Garthdee Road, Aberdeen, United Kingdom
j.clos@rgu.ac.uk

Abstract

Automatically classifying text documents is an active research challenge in document-oriented information systems. It allows such systems to help users browse massive amounts of data with ease by categorizing it on some axis of interest, detect the unknown authors of unsigned work, and analyze corpora along predefined dimensions of interest such as sentiment, emotion or stance. However, current approaches are biased towards building complex black-box algorithms focused on producing high accuracy predictions at the cost of not being able to explain the rationale behind their decisions. Lexicon-based classifiers offer a white-box alternative to these approaches by using a trivially interpretable additive model at the cost of classification accuracy. The contribution of this paper is RELEXNET, a computational architecture that models lexicons as naive gated recurrent networks, allowing us to train them using standard optimization approaches. We evaluate our approach on two tasks: stance detection and sentiment classification, and show that our approach is competitive with standard black-box shallow classifiers.

1 Introduction

Text classification is a core task in natural language processing, with applications ranging from web search to author detection. For example, support vector machines [Hearst *et al.*, 1998], a common family of classification algorithms [Fernández-Delgado *et al.*, 2014] have helped improve document navigation tasks by categorizing web search results [Chen and Dumais, 2000], analyzed corpora to identify anonymous authors [Diederich *et al.*, 2003], and are used to identify spam e-mails [Drucker *et al.*, 1999] at large scale. However, these supervised classification algorithms provide predictions with no means of explanation. Understanding the reason behind a classification allows us to establish trust in further predictions, which has far-reaching consequences in algorithms deployed in production systems such as search engines and document categorization pipelines. Lexicons fill this need by offering a trivially interpretable additive model, where the probability of an instance belonging to a class is modeled as a weighted sum of the probabilities of each term of that instance belonging to that class. An analyst can then examine the terms of an instance and their weights to understand the reason behind a prediction. However, current techniques used to build those lexicons are lacking in many respects compared to standard supervised text classifiers. This paper attempts to conciliate lexicon-based classification and traditional classification models by defining a simple and effective training procedure that can generate lexicons with a classification accuracy that is competitive with standard classification algorithms. We illustrate the explanation step of a lexicon prediction through an example in figure 1.

We first formalize the concept of lexicons and explore the state of the art in the domain of lexicon-based classification. We then detail our contribution, formalizing lexicon-based classification as a gated computational graph and inducing optimal weights using a regularized objective function. We then detail our evaluation protocol on two classification tasks: stance detection and sentiment classification. We perform an evaluation against standard lexicon learning techniques and baselines found in the literature and report that our approach significantly outperforms standard text classification techniques. Finally, we analyze and discuss our results, before exploring the next steps of our work.

2 Related works

Despite its widespread use in real-world applications, text classification heavily relies on black-box models offering little if any explanation on their predictions [Ribeiro *et al.*, 2016]. Lexicon-based classifiers overcome this limitation by constraining the classification to a simple model: each term has a score for each class, those scores get weighted according to the frequency of that term in the instance and then added together, and finally the class with the highest total score for a given instance is chosen as the prediction. Such a classification model offers transparency: each prediction can be explained trivially by analyzing the terms that were present in the text, and any domain expert could revise the model manually with a simple text editing software. This transparency however comes at the cost of some classification accuracy, due to the simplistic nature of its inference scheme.

Example 1. In a binary sentiment classification setting, for a given sentence "*I love horror books*", a lexicon $\mathcal{L}$ referred on the figure, the lexicon could find an aggregated score of $f(love) \times 1.0 + f(horror) \times 0.3 + f(books) \times 0.5 = 1.8$ for the *Positive* class, and $f(love) \times 0 + f(horror) \times 0.7 + f(books) \times 0.5 = 1.2$ for the *Negative* class, where f is a function measuring some notion of local term frequency. The decision function $\mathcal{D}$ would then return the class with the maximum value, i. e., *Positive*. A human reader can read the sentence and identify that the term "*love*" is responsible for tipping the classification towards the *Positive* class.

Example Lexicon

Term	*Positive*	*Negative*
love	1.0	0.0
horror	0.3	0.7
books	0.5	0.5

Figure 1: Classification and explanation with a sentiment lexicon

2.1 Lexicon-based classification

Lexicons are linguistic tools for classification and feature extraction [Clos *et al.*, 2017; Bandhakavi *et al.*, 2016]. They take the form of a list of terms weighted by their strength of association with a given class. Some lexicons also contain additional contextual information in order to help their users build more complex models [Muhammad *et al.*, 2016] as well as a list of terms which modify the strength (intensifiers, diminishers) or the polarity (negators) of other terms within a predetermined window. We describe the core of a lexicon as follows:

Formal lexicon. A lexicon Lex is a tuple $Lex = \langle \mathcal{L}, \mathcal{A}, \mathcal{D} \rangle$

where: $\mathcal{L}: \quad T \times C \mapsto \mathbb{R}$

$\mathcal{A}: \quad \mathbb{R}^n \mapsto \mathbb{R}$

$\mathcal{D}: \quad \mathbb{R}^n \mapsto \mathbb{R}$

For a given dictionary of terms T and set of classes of interest C, $\mathcal{L}$ is a mapping function that assigns an unbounded value to each pair (t, c) where term $t \in T$ and class $c \in C$. The function $\mathcal{A}$ is an aggregation function that accumulates scores and returns one value, and $\mathcal{D}$ is a decision function that selects and returns a single one of these aggregated values. Concretely, the mapping determines an evidence score for each term using a look-up list (the lexicon), propagates it to the aggregation function which aggregates the evidence into one cumulative score per class. Finally, the decision function evaluates each score to select the one that is the most likely.

This leads us to define a core challenge in lexicon-based classification: the lexicon induction problem. The next section reviews techniques traditionally used to solve the lexicon induction problem.

Lexicon induction problem. The lexicon induction problem is the estimation, given aggregation function $\mathcal{A}$ and decision function $\mathcal{D}$, of the optimal function $\mathcal{L}$ so that the resulting lexicon $Lex = \langle \mathcal{L}, \mathcal{A}, \mathcal{D} \rangle$ minimizes its classification errors on unseen data.

The added complexity in this paper stems from the presence of a new class of terms: **modifiers**. Modifiers are either intensifiers, diminishers or negators, and while they have no class value of their own they alter the value of class-bearing terms in a specific context window. Modifiers can be represented as a single value that is multiplied to the class value of all terms within a certain context window. For instance, the adverb "very" would have a modifier value of 1.3, meaning that a term such as "bad" which would have a negative value of 0.9 in a sentiment analysis setting would end up having a modified value of 1.17 after consideration. We note that modifier terms should ideally be able to interact with each other (e. g., "not very" being different from "not" and "very" applied separately) but that is left for future work.

2.2 Lexicon induction techniques

Research in lexicon induction outlines multiple families of techniques that can be used to generate a computational lexicon. Those techniques are either built on an extensive lexical resource such as an ontology, or on an estimation of strength of association between each term and a class in a reference corpus. Research has shown that merging multiple lexicons produces a reliable feature extractor to augment an existing classifier [Wang and Cardie, 2014], but using those lexicons for direct classification was not explored as it would blur the link between features and prediction.

Traditional hand-crafted lexicons (THCL) Due to the computational cost of building a lexicon from text, early lexicons were hand-crafted by domain experts [Stone *et al.*, 1966] and while higher performance in automated classification tasks has been shown using modern techniques, there still exist handcrafted lexicons in use to this day such as the Linguistic Inquiry and Word Count lexicon [Pennebaker *et al.*, 2001]. The strengths of these approaches are that they generalize well and are highly interpretable due to their human (and not algorithmic) origin. Conversely their weakness are that they tend to be small due to the human labor involved in generating them, and less effective than other methods due to their focus on human interpretability. However they can provide a commonsense knowledge back-up in hybrid lexicons [Muhammad *et al.*, 2014] with some degree of success.

Ontology-based lexicons (OBL) OBL learning techniques use a few human-provided seed words for which the class is known, and leverage some external relationship (typically synonymy, antonymy and hypernymy) in a semantic graph such as WordNet [Miller, 1995] to propagate class values along that graph [Esuli and Sebastiani, 2006]. Because this family of techniques is extremely foreign to the one we are proposing, we do not evaluate against it and only refer to it for the sake of exhaustiveness.

Corpus statistic-based lexicons (CSBL) CSBL learning techniques use a labeled corpus of interest in order to learn a domain-specific lexicon. The two main statistics used for

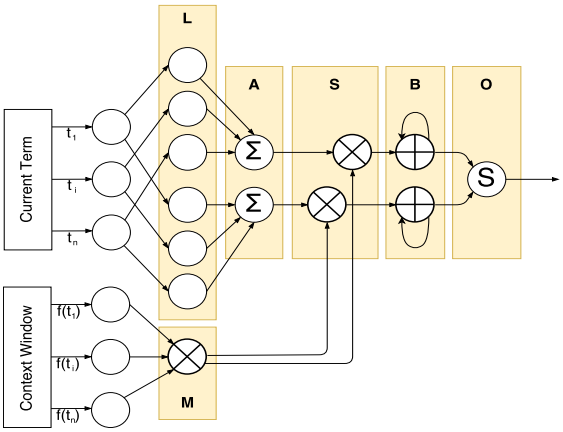

Figure 2: The RELEXNET topology

this purpose are the conditional probability (equation 1) of observing a term given a class, and the pointwise mutual information (PMI, equation 2) between the observation of a term and the observation of a class. These approaches are flawed in that they can overemphasize spurious correlations between terms and classes. For example, if a non-class specific term such as "Monday" accidentally co-occurs too often within one class, it will be misconstrued as being indicative of that class, and the lexicon will overfit. Bandhakavi et al. [Bandhakavi *et al.*, 2014] describe a method for building conditional probability-based lexicons and Turney [Turney, 2002] an approach using PMI and an external search engine to compute lexicon scores. Other works [Clos *et al.*, 2016] have shown some improvement using the normalized PMI measure (NPMI, equation 3) on a stance classification task.

$$P(t;c) = \frac{p(t|c)}{\sum_{i=0}^{|C|} p(t|c_i)} \tag{1}$$

$$PMI(t;c) = \frac{\log(p(t;c))}{p(t)p(c)} \tag{2}$$

$$NPMI(t;c) = \frac{\frac{\log(p(t;c))}{p(t)p(c)}}{-\log\left[p(t;c)\right]} \tag{3}$$

3 The RELEXNET architecture

In this section we present the RELEXNET architecture, presented in figure 2. The goal of RELEXNET is to jointly learn lexicon and modifier terms. We expect modifiers to be adverbs which alter the class valence of neighboring terms to different degrees, so in order to accelerate the learning we limited the sets of candidate terms to adverbs (e. g., very, etc.) and conjunctions (e. g., and, or, etc.). Each term is processed in their own timestep (one at a time), and the current score (aggregating the current timestep with the previous ones) is passed to the next timestep until the final term is processed and a prediction can be produced.

Figure 2 presents an architecture for binary classification. At the top of the diagram is the standard vocabulary layer (first layer), where each term of the instance is fed one at a time, activating the corresponding lexicon units (second layer, marked **L**) before being summed up in the aggregation unit (third layer, marked **A**), which is then modified by modifier terms nearby (fourth layer, marked **S**) and passed on to the next timestep (next term of the instance) via a summation gate (fifth layer, marked **B**). At the end of the sequence we examine the output of the Softmax function (sixth layer, marked **O**) which determines the probability distribution over classes. At the bottom of the diagram we process a context window of arbitrary size. For a context window of size c and for timestep i, all terms from t_{i-c} to t_{i+c} are fed as a bag of words as the context window ($f(t_i)$ denoting the frequency of term i in the context window). A weighted sum of those terms is then multiplied together before being fed in the top part of the network, to modify the score of the current term.

The forward pass of the graph can be represented in the following set of equations:

$$L_{i,j} = w_{i,j} \times f_1(t_i) \tag{4}$$

$$M = \prod_{i=1}^{|t|} (m_i \times f_2(t_i)) \tag{5}$$

$$A_j = \sum_{i=1}^{|t|} L_{i,j} \tag{6}$$

$$S_{i,j} = A_i \times M \tag{7}$$

$$B_{j,t} = B_{j,t-1} + \sum_{i=1}^{|c|} S_{i,j} \tag{8}$$

$$O_t = \bigoplus_{j=1}^{|c|} \frac{e^{-B_{j,t}}}{\sum_{m=1}^{|c|} e^{-B_{m,t}}} \tag{9}$$

In this equation $\bigoplus$ is a function that takes for input a sequence of real numbers and outputs a vector that contains them. The network passes on the data, and at each timestep feeds a one-hot encoding vector of the current term. We detail each part as follows:

- $L_{i,j}$ receives the one-hot encoding vector of the current term i for class j. In figure 2 we can observe that each term is mapped to N lexicon units, where N is the number of classes. It is formalized in equation 4.
- A_j sums the amount of evidence for class j. It is formalized in equation 6.
- M combines all modifiers present within the arbitrary word window into one modification score. It is formalized in equation 5.
- $S_{i,j}$ applies the modification score calculated in equation 5 to the score calculated in equation 6 for term i and class j. It is formalized in equation 7.
- $B_{j,t}$ communicates the output of $S_{i,j}$ at timestep t to the next timestep $t+1$. It is formalized in equation 8.

- O_t receives the output of $B_{j,t}$ (for all classes j) and outputs a probability distribution over the classes at timestep t. It is formalized in equation 9.

4 Experimental setup

4.1 Setting up RELEXNET

The training and testing of our approach was done using a 10-fold cross-validation while fixing the random seed of our algorithm to ensure that the results were consistent. The context size was arbitrarily fixed to the average phrase length in the corpora and not optimized further

Since 3 of the 4 datasets are balanced, we selected classification accuracy as our performance measure.

4.2 Datasets

We performed our evaluation on two tasks, containing a total of 4 datasets for stance classification and sentiment analysis. We describe the tasks and datasets associated in the rest of this section:

- **Stance detection** is the study of local stance of a document with respect to a topic or another stance. For example, if the topic of discussion is "death penalty" and a document d_1 is for the death penalty, then a document d_2 that is against the death penalty is said to be in disagreement with document d_1, while a document d_3 that is also for the death penalty is said to be in agreement with document d_1. In this work, we consider a reduced version of stance classification where the topic is not observed, and the classifiers are not provided with context.
  - **The IAC dataset** is a subset of the Internet Argument Corpus [Walker *et al.*, 2012] containing forum comments crawled from 4FORUMS on different topics: e. g., politics, ... and labeled on a scale from -5 to 5. A subset of comments that ensured disjoint class membership (with an average score far from 0) and containing more than 3 words was binned into 2 classes (agreement and disagreement) and used for our experiments.
  - **The CD dataset** is a dataset collected from the CREATEDEBATE forum dedicated to social argumentation on political and religious topics and labeled using 2 classes (agreement and disagreement). The dataset was used as is with no processing.
- **Sentiment classification** is the study of the sentiment (positive or negative) contained within a piece of text. While many datasets propose finer-grained sentiment classes (including neutral class or numerical sentiment score) we chose to use a binary classification task as our goal.
  - **The AYI dataset** was collected from Amazon, Yelp and IMDB and was built from individual sentences from product, location and movie reviews (respectively).
  - **The AMZ dataset** was collected from Amazon user reviews.

Stance classification			
Baseline lexicons	CPBLEX	0.524	0.441
	PMILEX	0.557	0.529
Baseline classifiers	NAIVEBAYES	0.536	0.474
	SVM	0.589	0.594
	DECISIONTREE	0.582	0.573
Approach	**RELEXNET**	**0.655***	**0.677***
Sentiment classification			
Baseline lexicons	CPBLEX	0.528	0.519
	PMILEX	0.571	0.554
Baseline classifiers	NAIVEBAYES	0.665	0.637
	SVM	0.689	0.671
	DECISIONTREE	0.742	0.707
Approach	**RELEXNET**	**0.751**	**0.719***

Figure 3: Experimental results

4.3 Baselines

We used two families of baselines as comparison points with our approach:

Lexicons: Two lexicons used as a baseline are the CPBLEX and the PMILEX, which are standard methods for building lexicons for other purposes, such as sentiment lexicons [Jurafsky and Martin, 2016]. Section 2.2 on corpus-based lexicons details their implementation ;

Standard classifiers: SVM (with a RBF kernel), which has been shown to perform well in stance detection tasks by Yin et al. [Yin *et al.*, 2012] and is a regular top performer in general classification tasks [Fernández-Delgado *et al.*, 2014], and NAIVEBAYES and DECISIONTREE which are two popular baselines for text classification. Parameters for the classifiers were taken from the default recommendations of the SCIKIT-LEARN [Pedregosa *et al.*, 2011] library.

5 Results and discussion

Table 3 shows that RELEXNET significantly outperforms the baselines (on a two-tailed paired T-test, with $p < 0.05$), while producing a human-readable lexicon that can be used to explain predictions. The hyperparameters were empirically determined on a hold-out set, leading to the choice of a lexicon size of 400 words (selected by corpus frequency), a regularization coefficient of 0.5 and a momentum velocity of 0.7.

6 Conclusion

In this work we showed the viability of using regularized backpropagation to efficiently learn effective lexicon weights, producing a lexicon that is competitive with standard classifiers and outperforms baseline techniques such as SVM. Our future works will focus on improving the performance of RELEXNET by taking into account modifier phrases, which are built from multiple modifier terms (e. g., "not very") and have a modifier valence of their own.

References

[Bandhakavi *et al.*, 2014] Anil Bandhakavi, Nirmalie Wiratunga, P Deepak, and Stewart Massie. Generating a word-emotion lexicon from #emotional tweets. In *Proc of the 3rd Joint Conf. on Lexical and Comp. Sem.*, 2014.

[Bandhakavi *et al.*, 2016] Anil Bandhakavi, Nirmalie Wiratunga, P Deepak, and Stewart Massie. Lexicon based feature extraction for emotion text classification. *Pattern Recognition Letters*, 2016.

[Chen and Dumais, 2000] Hao Chen and Susan Dumais. Bringing order to the web: Automatically categorizing search results. pages 145–152, 2000.

[Clos *et al.*, 2016] Jérémie Clos, Nirmalie Wiratunga, Stewart Massie, and Guillaume Cabanac. Shallow techniques for argument mining. In *ECA'15: Proceedings of the ECA*, volume 63, page 2, 2016.

[Clos *et al.*, 2017] Jérémie Clos, Anil Bandhakavi, Nirmalie Wiratunga, and Guillaume Cabanac. Predicting emotional reaction in social networks. In *European Conference on Information Retrieval*, pages 527–533. Springer, 2017.

[Diederich *et al.*, 2003] Joachim Diederich, Jorg Kindermann, Edda Leopold, and Gerhard Paass. Authorship attribution with support vector machines. *Applied intelligence*, 19(1):109–123, 2003.

[Drucker *et al.*, 1999] Harris Drucker, Donghui Wu, and Vladimir N Vapnik. Support vector machines for spam categorization. *IEEE Transactions on Neural networks*, 10(5):1048–1054, 1999.

[Esuli and Sebastiani, 2006] Andrea Esuli and Fabrizio Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422. Citeseer, 2006.

[Fernández-Delgado *et al.*, 2014] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems. *J. Mach. Learn. Res*, 15(1):3133–3181, 2014.

[Hearst *et al.*, 1998] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, 1998.

[Jurafsky and Martin, 2016] Dan Jurafsky and James H Martin. *Lexicons for Sentiment Extraction*, chapter 18. Pearson, 2016.

[Miller, 1995] George A Miller. Wordnet: a lexical db for english. *Comm. of the ACM*, 38(11):39–41, 1995.

[Muhammad *et al.*, 2014] Aminu Muhammad, Nirmalie Wiratunga, and Robert Lothian. A hybrid sentiment lexicon for social media mining. In *Tools with AI (ICTAI), IEEE 26th International Conf. on*, pages 461–468, 2014.

[Muhammad *et al.*, 2016] Aminu Muhammad, Nirmalie Wiratunga, and Robert Lothian. Contextual sentiment analysis for social media genres. *Knowledge-Based Systems*, 108:92–101, 2016.

[Pedregosa *et al.*, 2011] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

[Pennebaker *et al.*, 2001] James W Pennebaker, Martha E Francis, and Roger J Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71:2001, 2001.

[Ribeiro *et al.*, 2016] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016.

[Stone *et al.*, 1966] Philip J Stone, Dexter C Dunphy, and Marshall S Smith. *The general inquirer: A computer approach to content analysis.* MIT press, 1966.

[Turney, 2002] Peter D Turney. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proc. of the 40th annual meeting on ACL*. ACL, 2002.

[Walker *et al.*, 2012] Marilyn A Walker, Jean E Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. A corpus for research on deliberation and debate. In *LREC*, pages 812–817, 2012.

[Wang and Cardie, 2014] Lu Wang and Claire Cardie. Improving agreement and disagreement identification in online discussions with a socially-tuned sentiment lexicon. *ACL 2014*, page 97, 2014.

[Yin *et al.*, 2012] Jie Yin, Paul Thomas, Nalin Narang, and Cecile Paris. Unifying local and global agreement and disagreement classification in online debates. In *Proc. of the 3rd Workshop in Comp. Approaches to Subjectivity and Sentiment Analysis*, pages 61–69. ACL, 2012.