Domain-Specific Sentiment Lexicons Induced from Labeled Documents

Gabriele Rosi
Politecnico di Torino
s291082@studenti.polito.it

Andrea Tampellini
Politecnico di Torino
s288266@studenti.polito.it

***Abstract*—Sentiment analysis is one of the most important NLP related tasks. In recent years, sentiment lexicons of limited size that include generic polarity scores were created. In this work, we provide a neural network based method that expands the coverage of these types of lexicons to a very large vocabulary. To accomplish this task, at first we relied on a linear model to induce a set of 12 domain-specific sentiment lexicons, then we train a deep neural model on word vector representations to expand the generated lexicons. Finally, we tested the performance of our trained models on 5 different datasets related to 5 different domains. Our experiments show that training the models on domains that match the dataset domains yields better results in terms of accuracy. The code is available at the following link: https://github.com/Gabrysse/SentimentLexiconExpansionModel.**

I. Problem statement

One of the most prominent tasks in the natural language processing field is sentiment analysis. It has many applications that are mainly related to analyzing the public sentiment in different settings, such as social media posts or e-commerce reviews and recommendations. One of the easiest to follow approaches to perform this task exploits sentiment lexicons, that can be run out-of-the-box as they don't need labeled training data. More specifically, a sentiment lexicon is a vocabulary that for each word w_i provides a label l_i that describes the word sentiment polarity. Depending on the lexicon, the labels can be a set of classes such as {`positive, negative`} or {`positive, neutral, negative`}, or it could be a continuous value in a specific range such as [-1, 1], where the lowest values represent negative sentiment and vice versa. In the latter case, the strength of a sentiment can be better expressed by the continuous value. For example, the word *amazing* would have a higher polarity score than the word *good*, even if both of them are expressing positive sentiments. While these sentiment lexicons have positive aspects, they also present shortcomings, such as the following two:

1) Since lexicons are based on domain and context independent polarities, the sentiment scores can be highly dependent on the domain where the words appear. For example, the word *hit* in the musical domain would have a positive polarity, as the meaning is a song that is having a great success. In other domains, the word *hit* could be seen as negative, as the meaning could be to beat or punch someone.
2) Sentiment lexicons are usually created manually and have relatively small vocabulary sizes. For example, the VADER lexicon [1], used in our experiments, includes the polarity scores for 7506 words.

The approach described in the following sections is aimed at mitigating the aforementioned problems. At first, domain-specific sentiment lexicons are automatically generated with a data-driven approach. Then, the lexicons coverage is extended using large-scale vector representations and a deep neural regression model. While this approach doesn't address the issues of context and polysemy, the conducted experiments show that there are significant difference between domains.

II. Methodology

The following approach can be divided in two steps. In the first step, labeled datasets of different domains are exploited to induce seed data for a limited set of words included in each domain. In the second step, a model based on a neural network is trained to learn the polarities of a much larger vocabulary with the help of word vector representations.

A. Tokenization

The preprocessing phase of the adopted approach is straightforward. For each document in a domain, a list of token is generated by splitting the lowercased text at each non-alphanumeric character, with the only exception of negated words. To detect negations, first the words ending with *"n't"* are transformed into the non negated word followed by "not". Then, during the tokenization process, every occurrence of *"not"* followed by another word is considered a single token that's different from the token that represents the non-negated word.

B. Seed Data Induction

In the first step, seed data is generated for n domains using domain-specific document sets $D_i \in X \times Y (i = 1, ..., n)$, labeled with sentiment polarity labels in $Y = \{positive, negative\}$. In particular, a set of features $F_i = V_i \cup \{\overline{w_j} \,|\, w_j \in V_i\}$ is defined for each D_i, where V_i is the vocabulary of D_i, w_j is a word and $\overline{w_j}$ is the negated version of that word. Then, the documents x_j in a domain D_i are mapped to term frequency-based document vectors x_j in feature space F_i. These features are used to train a linear model:

$$f_i(x) = (Y - X\beta)^T (Y - X\beta) + \lambda \beta^T \beta \tag{1}$$

The final polarity score for each word w_j in V_i is its resulting linear coefficient weight $w_{i,j}$ obtained from the trained model. While the negated words are considered to train the linear model, they are only used as a way to reduce the noise that could affect the non-negated word features. Furthermore, words that rarely appear in the datasets are also disregarded, because their final score could be too unreliable. In particular, a minimum threshold f_{min} of 500 words was set to filter the words, which results in the training data for the next step that can be defined as:

$$T_i = \left\{ (w_j, w_{i,j}) \,|\, w_j \in V_i, \sum_{(x,y)\in D_i} f(x, w_j) \geq f_{min} \right\} \quad (2)$$

C. Neural Vector-based expansion

The main goal of the second phase is to extend the words covered by the induced seed data to a significantly larger vocabulary. Word vector representations trained on large corpuses [2] [3] can capture important aspects of lexical semantics, and they can also reveal sentiment signals [4]. In this step, vectors representations of the words covered by the seed data induced, as described in Section II-B, are used to train a deep neural network based model $\phi_i(\mathbf{v}_w) \in \mathbb{R}$. After the training, the deep neural model can be used to predict the polarity scores of the words that are not covered by the induced seed data, de facto extending the sentiment vocabulary.

The neural network architecture is described in Figure 1. It is composed by several hidden layers followed by a batch normalization layer, a ReLU activation function and a dropout layer. The penultimate layer, instead, is only followed by a batch normalization layer and a softmax activation function. Finally, the last layer is simply a fully connected layer, but a scaling of the softmax scores to the sentiment score range observed in the training data is also performed. This technique proves beneficial to the overall results.

III. Experiments

In this section, we are going to describe the experimental results obtained using our method. Our experiments consist of three steps:

- First, we induce a polarity score for each word based on the review sentiment in which it appears
- Then we train our neural network to increase the word coverage and learn more domain-specific sentiment scores
- Finally, we assess the performance of each model testing it on different domains

As input corpus, we used the Amazon review dataset [5], a dataset composed by 233.1 million review gathered from amazon.com in a period spanning from May 1996 to October 2018. Each review is characterized by a rating from 1 to 5. We considered as positive the reviews with $rating > 3$ and negative those with $rating < 3$. The reviews with $rating = 3$ have been discarded. Furthermore, we assigned the label -1 to the negative reviews while the label 1 to positive ones. We

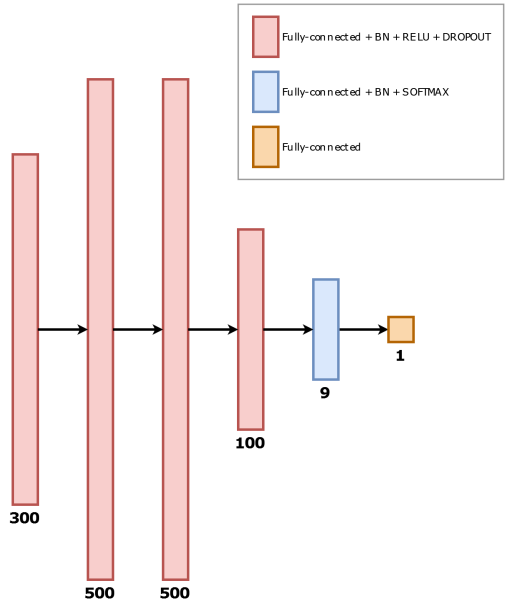

Fig. 1. Neural network architecture

selected a subset of 12 domains from the publicly available repository[1] based also on the ones considered in [6].

Polarity score induction. To obtain our seed data, we decided to use a regression model for binary classification. To save computation time, we chose a Ridge regression model instead of SVM as in [6].

Neural network training. In each experiment our neural network has been trained for 100 epochs with a batch size of 32, dropout rate of 20%, Adam optimizer with initial LR of 0.001, dynamic learning rate schedule (halving after 4 epochs of validation loss stagnation) and early stopping (stop after 12 epochs of no validation loss improvements).

A. Domain generic and domain specific sentiment scores

To validate the performance of the proposed approach, first we created a domain generic prediction system. This model relies on GloVe CommonCrawl embeddings [3] as word vectors and on VADER lexicon [1] as ground truth sentiment score. We eliminated every word in VADER that is not present in GloVe. A stratified split of 60/20/20 was used in order to create train/validation/test portion with equally distributed sentiment scores.

Similarly, the domain specific prediction system follows the same structure, with the only difference that our seed data has been used as ground truth.

Finally, we trained 12 domain specific models, one for each of the chosen domains. Then, we measured the Pearson correlation between the domain specific sentiment score obtained from our model and the complete VADER lexicon. Any word that wasn't covered by our model was assumed to have a neutral polarity score of 0.0. We decided to compare our review time period (May 1996 - October 2018) with the

[1] https://nijianmo.github.io/amazon/index.html

one analyzed in [6] (May 1996 - July 2014). In Figure 2 the correlation comparison is reported.

As we can see, the correlation coefficients between the domain specific sentiment scores obtained from the neural models and the complete VADER lexicons are significantly higher than the correlation coefficients between the induce seed data polarities and VADER in most cases. In general, it's not expected for domain specific sentiment scores to correlate well with VADER polarities, as the domain specific scores should differ from generic ones. The higher correlation coefficients are mainly a result of the better word coverage of the trained neural model, which is able to predict the scores of a large number of words that otherwise would have received a default polarity score of 0. For a similar reason, the seed-VADER correlation increases when the seed data are induced from a larger number of reviews, which provides a better word coverage. Instead, the correlation between VADER and the model predictions don't seem to improve when adding more training data, with the coefficients increasing for some domains and decreasing for others.

B. Unsupervised sentiment classification

To assess the performance of our domain specific sentiment prediction models, we performed five different tests on the following datasets:

- **IMDB movie review dataset**[2] [7] containing 25k positive review and 25k negative review obtained from IMDb. For the preprocessing, we simply assigned the label -1 to the negative reviews and the label 1 to the positive ones.
- **Hotel review dataset**[3] containing 515k reviews scraped from Booking.com. Each review is characterized by a negative review, a positive review, a score and other information that we discard. We simply concatenate the positive review (if present) and the negative review (if present) and we assign the label -1 to the reviews with $score < 7$ while the label 1 to the ones with $score \geq 7$.
- **Fake and real news dataset**[4] [8] consisting of 39k news marked as *fake* or *real*. Here we assign the label -1 to the fake news while the label 1 to real ones.
- **Coronavirus tweets NLP dataset**[5] containing 41k tweet gathered from Twitter regarding coronavirus (COVID-19). The original dataset provides 5 types of labels: *Extremely Positive*, *Positive*, *Neutral*, *Negative*, *Extremely Negative*. We simply discard the *Neutral* tweet, and we assign the label -1 to *Extremely Negative* and *Negative* tweets, while the label 1 to *Positive* and *Extremely Positive* ones.
- **Spam Text Message Classification dataset**[6] containing 5k spam messages. Each message is labeled as spam or ham; we simply assign the label -1 to the spam messages, while the label 1 to ham ones.

The 12 domain specific models previously trained on the reviews up to 2014 were used to predict the scores of these datasets entries. For each document in the datasets, the score was calculated as the average polarity score of the tokens. Formally:

$$f(x) = \frac{1}{|x|} \sum_{i=0}^{|x|} \phi(\vec{v}_{x_i}) \tag{3}$$

where $|x|$ is the document length, x_i is the i-th word in x and $\phi(\vec{v}_{x_i})$ is the neural prediction model given the word vector for w. The documents that got a positive score were classified as positive, the others were classified as negative. The accuracy obtained with each domain is reported in Table I.

Next, for each test dataset, we selected the best Amazon review dataset, based on the achieved accuracy, and we analyzed the output of each model on the two time periods cited before (May 1996 - July 2014 and May 1996 - October 2018) to find the most positive and negative words. These words are reported in Table II.

As reported in Table I, the best results were achieved on the IMDb review dataset with a score of 0.859, followed by the Hotel review dataset with a score of 0.794. The domain that reached the top performance on the IMDb dataset is Movies and TV, as it was to be expected considering that IMDb matches this domain. The Hotel review dataset obtained its best score with the Grocery and Gourmet food domain. While the domain is not perfectly matching, there are still similarities considering that hotels can also provide dining services.

The Coronavirus tweets dataset, instead, didn't get a high score, reaching a score of 0.6132 with the top domain Automotive. This does not come as a surprise because none of the domains used is related to the medical field. Also, the Amazon datasets used to generate seed data only included reviews until 2014, which is years before the Coronavirus outbreak. For both of these reasons, it's reasonable to believe that the model was not able to learn the polarities specific to the Coronavirus domain.

A particular attention should be given to Fake news and Spam messages: these two are not sentiment analysis oriented datasets. In fact, we were not expecting to get good results on these datasets as the text classification task is not based on sentiment, but rather on other semantic aspects. In any case, they both obtained an accuracy score of over the 60% and it cannot be considered a bad result.

An example of how the model can adapt to different domains can be found in the word *rotk*, which is an abbreviation of *Return of the King*, the last movie of the trilogy *The Lord of the Rings*. This movie is considered by many to be a masterpiece. With a score of 9 on 10 on imdb.com and a Tomatometer score of 93% on rottentomatoes.com, it's not a surprise that the polarity of this word (0.12) is positive in a domain related to movies. On the contrary, the word tends to

[2]https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

[3]https://www.kaggle.com/datasets/jiashenliu/515k-hotel-reviews-data-in-europe

[4]https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset

[5]https://www.kaggle.com/datasets/datatattle/covid-19-nlp-text-classification

[6]https://www.kaggle.com/datasets/team-ai/spam-text-message-classification

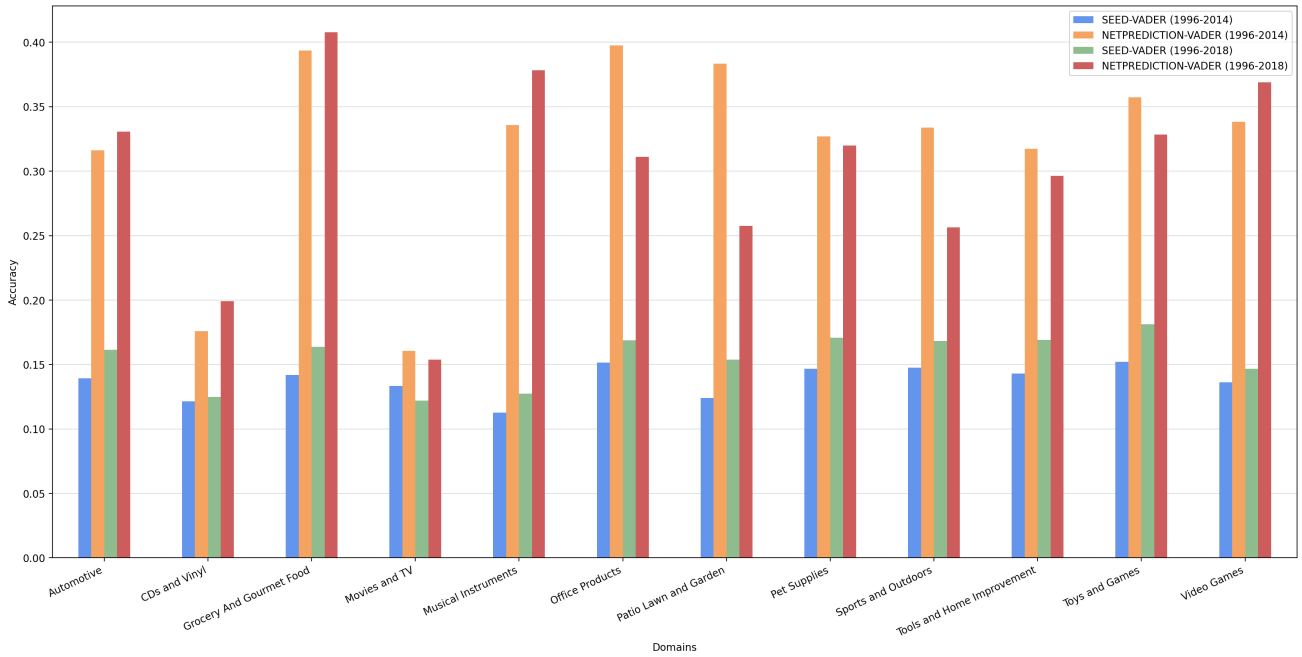

Fig. 2. Pearson correlation Seed data-VADER and predicted score-VADER in two different time period (May 1996-July 2014 and May 1996-October 2018)

be neutral in the other domains, such as Automotive (-0.04), Grocery and Gourmet Food (0.04) or Musical Instruments (0.05), where it's not expected to have a particular meaning.

An important aspect we can observe from Table II is that our model is able to capture the true sentiment score of words with spelling errors (e.g. *dissapointment*, *discusting*, *efficeint*). The word vector representations of words with spelling errors is similar to the representation of the correct word, meaning that the two words will have similar polarity scores. This proves the effectiveness of training the neural model to expand the vocabulary to a very large number of words.

One last notable aspect from Table II is how the absolute polarity scores increase when the model is trained with the reviews up to 2018, instead of the reviews up to 2014. It's possible to see that on every test, the negative polarities decreased when training on more years, whereas the positive polarities increased. We believe that this behavior can be associated to the fact that with a large number of reviews taken into consideration, words that were already negative and positive strengthened their orientation towards a negative or a positive sentiment.

IV. Conclusions

In this work, a neural based approach to sentiment analysis has been proposed. As seen in Section III our experiments confirmed the ability of the model to infer sentiment score of unseen words. The accuracy results obtained also show that the model performs significantly better with domains that refer to the same context (e.g. IMDb dataset and Movies and TV Amazon review dataset), while the performance decreases with out-of-the-scope domains such as Coronavirus tweet dataset or Fake and real news dataset.

To improve our model, it could be interesting to analyze other techniques to better identify negations or to exploit different word vector representation models, even those that depend on context (e.g. BERT or ELMo embeddings).

References

[1] C. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 8, no. 1, pp. 216–225, May 2014.

[2] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[3] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[4] S. Rothe, S. Ebert, and H. Schütze, "Ultradense word embeddings by orthogonal transformation," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2016.

[5] J. Ni, J. Li, and J. McAuley, "Justifying recommendations using distantly-labeled reviews and fine-grained aspects," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 188–197.

[6] S. M. Islam, X. Dong, and G. de Melo, "Domain-specific sentiment lexicons induced from labeled documents," in *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 6576–6587.

[7] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150.

[8] H. Ahmed, I. Traore, and S. Saad, "Detecting opinion spams and fake news using text classification," *Security and Privacy*, vol. 1, p. e9, 12 2017.

Dataset	IMDb review	Hotel review	Fake news	Coronavirus tweet	Spam messages
Automotive	0.75154	0.758949	0.504823	**0.613204**	0.465529
CDs and Vinyl	0.71530	0.722683	0.536878	0.535103	0.439497
Grocery and Gourmet Food	0.73120	**0.794689**	0.502021	0.608659	**0.643806**
Movies and TV	**0.85092**	0.794031	0.604468	0.594427	0.630521
Musical Instruments	0.71608	0.614856	**0.616508**	0.589074	0.482764
Office Products	0.66418	0.765922	0.508685	0.555705	0.373070
Patio Lawn and Garden	0.74746	0.743610	0.546343	0.549934	0.472351
Sports and Outdoors	0.77830	0.773319	0.598165	0.593559	0.517235
Tools and Home Improvement	0.61606	0.649956	0.510357	0.536777	0.420646
Toys and Games	0.70886	0.705073	0.519101	0.584798	0.449013
Video Games	0.78394	0.688604	0.514898	0.547901	0.573968

TABLE I
ACCURACY OF UNSUPERVISED SENTIMENT CLASSIFICATION

	Top 5 Negative words				**Top 5 Positive words**			
	1996-2014		1996-2018		1996-2014		1996-2018	
IMDb review	*disapointing*	-0.2682	*disapointing*	-0.3327	*drawbacks*	0.1290	*wiseau*	0.1723
	refund	-0.2680	*stunk*	-0.3320	*downsides*	0.1281	*farrelly*	0.1582
	waste	-0.2676	*disapointment*	-0.3320	*gripes*	0.1280	*excelent*	0.1575
	poorest	-0.2669	*meh*	-0.3318	*complaint*	0.1246	*stomachs*	0.1468
	disapointed	-0.2666	*bleh*	-0.3316	*rotk*	0.1209	*exellent*	0.1422
Hotel review	*underwhelming*	-0.2590	*ripoff*	-0.4410	*comfiest*	0.0791	*padding*	0.1904
	disappointing	-0.2587	*discusting*	-0.4402	*briljant*	0.0790	*negatives*	0.1730
	uninspiring	-0.2586	*scam*	-0.4386	*thourghly*	0.0788	*excelente*	0.1664
	unwatchable	-0.2584	*disguisting*	-0.4368	*inkeeping*	0.0787	*complaint*	0.1652
	lackluster	-0.2583	*dissapointment*	-0.4367	*efficeint*	0.0787	*excelent*	0.1641
Fake news	*disappointing*	-0.1470	*doa*	-0.3037	*complaint*	0.1471	*gracias*	0.2207
	returned	-0.1470	*died*	-0.3034	*shutout*	0.1391	*encanto*	0.2001
	returning	-0.1470	*massimino*	-0.3033	*remits*	0.1391	*exellent*	0.1912
	useless	-0.1469	*unusable*	-0.3031	*milken*	0.1309	*thankyou*	0.1594
	miserable	-0.1469	*worthless*	-0.3024	*planemakers*	0.1286	*siempre*	0.1590
Coronavirus tweet	*disappointing*	-0.2819	*unsatisfied*	-0.3545	*fedex*	0.1204	*excelent*	0.2204
	useless	-0.2787	*quit*	-0.3542	*pleased*	0.1179	*bueno*	0.2052
	dismal	-0.2778	*defeats*	-0.3539	*eliminated*	0.1159	*gr8*	0.2046
	unhappy	-0.2772	*waste*	-0.3537	*voila*	0.1139	*gracias*	0.1953
	cheaply	-0.2771	*worthless*	-0.3528	*survived*	0.1132	*goodjob*	0.1903
Spam messages	*worthless*	-0.2580	*died*	-0.4366	*luvd*	0.0779	*complaint*	0.1652
	worst	-0.2562	*shattered*	-0.4301	*def*	0.0762	*gr8*	0.1584
	disappointment	-0.2561	*worthless*	-0.4301	*addicted*	0.0760	*sue*	0.1559
	horrible	-0.2556	*urgh*	-0.4265	*excellent*	0.0754	*awesome*	0.1486
	threw	-0.2556	*worst*	-0.4260	*comfey*	0.0752	*answr*	0.1474

TABLE II
NEGATIVE AND POSITIVE WORDS LIST OVER THE TWO TIME PERIOD FOR EACH TEST DOMAIN