Emotion analysis in dataset

Meriem Aggoune
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

Lauri Heikka
Faculty of Information Technology and Electrical Engineering
Univ. of Oulu
Oulu, Finland
lauri.heikka@student.oulu.fi

Anusha Ihalapathiranae
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—We document a series of experiments on using a variety of natural language processing methods to classify emotions conveyed by sentences. In particular, we compare word categorizations, semantic similarity approaches and machine learning approaches. The machine learning approaches consist of classification models based on bag-of-words representations and convolutional neural networks applied on word embeddings. Our results indicate that machine learning approaches outperform string matching and semantic similarity with a considerable margin. Within the machine learning approaches, convolutional neural networks with word embeddings provide an improvement in accuracy of around 3-5 percentage points over bag-of-words models. The best accuracy on a hold-out validation set, around 93%, is achieved with a CNN approach using pre-trained word2vec embeddings. However, custom-trained word embeddings provide similar performance with less computational overhead.**

***Index Terms*—natural language processing, text classification, sentiment analysis**

I. Group Info

Group 10.

Members: Meriem Aggoune, Lauri Heikka, Anusha Ihalapathiranae

Title: Emotion Analysis in Dataset

Github: https://github.com/MeriemLil/NLP_Project

II. Introduction

Introduction here...

III. Data and Methodologies

A. Datasets and sources

Our main dataset of interest is a labeled dataset of 20 000 sentences with between 3 and 66 words. The dataset is provided by [1]. The sentences are labeled to convey one of six different emotions: fear, anger, joy, love, surprise and sadness. The dataset is collected following the approach of [2]. The sentences are tweets, and the emotion labels are hashtags at the end of the tweet. The sentences are preprocessed in that they contain no punctuation or special characters and all words are lowercase.

The dataset has a predefined split into training, test and validation samples, with 16 000, 2 000 and 2 000, respectively. Because the semantic similarity and string matching approaches do not employ any predictive modeling, in these sections we use the all 20 000 sentences to measure prediction accuracy. For the machine learning approaches, we use the training set for model training, the test set for model selection and hyperparameter tuning, and reserve the validation dataset as a final benchmark, to ensure a fair comparison accross approaches.

In addition, we use pre-trained, 300-dimensional word2vec embeddings from [3] and pre-trained fastText embeddings from [4]. Both datasets consist of 300-dimensional word embeddings, with embeddings for 3 and 1 million english words, respectively. For the string matching, we use the Harvard General Inquirer dataset, which contains categorizations for around 11 000 english words.

1) Harvard Inquirer and String Matching: The first method we explore...

2) Empath Client Categories: asdf

3) Semantic Similarity: asd

4) SentiStrength sentiment scores: As part of the project specification, we also use the SentiStrength client [5], to compute sentiment scores. These can be potentially be used as an additional feature in any approach to distinguish between the negative and positive emotions.

B. Bag-of-Words Models

asd

C. Convolutional Neural Networks

We replicate the convolutional neural network architecture used in [6]. The architecture is outlined in 1. Sentences are represented as $n \times k$ matrices of word vectors, where n refers to the length of the sentences and k to the dimension of the word vectors. All sentences are padded to length n.

In [6], pre-trained Word2Vec embeddings from [3] [4] are used. In addition to replicating this approach, we consider a

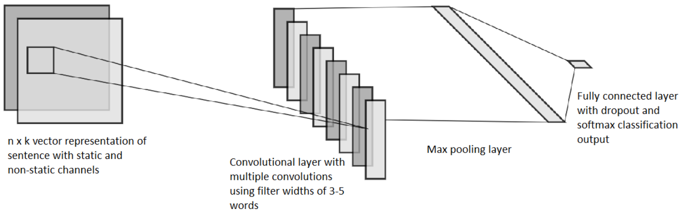

Fig. 1. Example of a figure caption.

few alternatives. First, we test pretrained fastText embeddings . Second, we explore explore training our own word2vec and fastText embeddings. The word2vec model... The fast-text model... For both models, we use a maximum distance between predicted words of five, and use early stopping on a word analogy evaluation task.

We experiment with 50-, 100- and 300-dimensional word embeddings for the custom-trained word embeddings. With the pre-trained embeddings, we are restricted to the commonly available 300-dimensional word embeddings.

IV. Results and Discussion

A. Model selection for bag-of-words

Table IV reports the accuracies on the test dataset used for model selection.

TABLE I
MAXIMUM ACCURACY ON THE TEST DATASET ACROSS DIFFERENT BAG-OF-WORDS SETUPS

parameter	*value*	*score*
vectorizer	CountVectorizer	0.886
	TfidfVectorizer	0.888
stopwords	nltk	0.888
	none	0.878
	sk	0.884
lemmatize	0	0.888
	1	0.886
max features	100	0.396
	500	0.728
	1000	0.865
	2000	0.880
	3000	0.886
classifier	DecisionTreeClassifier	0.866
	GradientBoostingClassifier	0.849
	LogisticRegression	0.878
	MultinomialNB	0.859
	RandomForestClassifier	0.888
	SVC	0.876

B. Validation set performance of best models

Table II reports the accuracies on the test dataset used for model selection.

TABLE II
VALIDATION AND TEST SET ACCURACIES FOR BEST MODELS

Classifier	Accuracy	
	Validation	*Test*
MultinomialNB	0.688	0.774
LogisticRegression	0.881	0.946
RandomForestClassifier	0.897	0.995
cnn fasttext	0.925	0.928
cnn own	0.92	0.922
cnn word2vec	0.929	0.927

TABLE III
VALIDATION PRECISION AND RECALL FOR BEST MODELS

Classifier	Metric	
	Precision	*Recall*
MultinomialNB	0.739	0.688
LogisticRegression	0.881	0.881
RandomForestClassifier	0.897	0.897
cnn fasttext	0.926	0.925
cnn own	0.92	0.921
cnn word2vec	0.929	0.93

V. Overall discussion and related literature

...

As mentioned in the introduction, the state of the art for text classification tasks has moved beyond word level embedding representations considered here. For example, [7], [8] and [9] employ pre-trained bidirectional encoder representations to achieve state-of-the-art results in sentence classification. However, constructing a replication and a fair comparison of these models is beyond the scope of this study. [1] Exploring alternative setups for training the word vectors is likewise beyond the scope of this paper.

The pretrained embeddings are quite large in size, while custom embeddings trained on the emotions dataset are much smaller. Furthermore, because model performance with embeddings trained on the emotions dataset is very close in performance to the pretrained embeddings, the custom models provide a lightweight alternative for model serving. It is important to note, however, that the emotions dataset contains very homogenous sentences across the training, test and validation datasets. This causes the vocabulary of the custom word embeddings to be much smaller than the pretrained word embeddings. Therefore the model is more likely to suffer in performance on out-of-distribution sentences.

...

VI. Conclusion

conclude.

VII. Latex examples

To be removed.

A. Equations example

$$a + b = \gamma \tag{1}$$

Use "(1)", not "Eq. (1)" or "equation (1)", except at the beginning of a sentence: "Equation (1) is . . ."

[1]There is a minimal, unofficial implementation of BERT [7] embeddings on the dataset we study [1] available at Kaggle by the dataset author. This implementation appears to exceed the accuracy of the best studied CNN model by around 0.3 percentage units.

B. Some Common Mistakes

- The word "data" is plural, not singular.
- In your paper title, if the words "that uses" can accurately replace the word "using", capitalize the "u"; if not, keep using lower-cased.
- Be aware of the different meanings of the homophones "affect" and "effect", "complement" and "compliment", "discreet" and "discrete", "principal" and "principle".
- Do not confuse "imply" and "infer".
- The prefix "non" is not a word; it should be joined to the word it modifies, usually without a hyphen.
- There is no period after the "et" in the Latin abbreviation "et al.".
- The abbreviation "i.e." means "that is", and the abbreviation "e.g." means "for example".

Acknowledgment

We thank Mourad Oussalah and other seminar participants for their helpful comments during the Natural Language Processing and Text Mining course project seminar.

References

[1] P. Govi, "Emotions dataset for nlp," 2020. [Online]. Available: https://www.kaggle.com/praveengovi/emotions-dataset-for-nlp

[2] E. Saravia, H.-C. T. Liu, Y.-H. Huang, J. Wu, and Y.-S. Chen, "CARER: Contextualized affect representations for emotion recognition," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 3687–3697. [Online]. Available: https://www.aclweb.org/anthology/D18-1404

[3] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," 2013.

[4] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[5] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas, "Sentiment strength detection in short informal text," *Journal of the American Society for Information Science and Technology*, vol. 61, no. 12, pp. 2544–2558, 2010. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/asi.21416

[6] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1746–1751. [Online]. Available: https://www.aclweb.org/anthology/D14-1181

[7] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. [Online]. Available: http://arxiv.org/abs/1810.04805

[8] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *CoRR*, vol. abs/1906.08237, 2019. [Online]. Available: http://arxiv.org/abs/1906.08237

[9] C. Sun, X. Qiu, Y. Xu, and X. Huang, "How to fine-tune BERT for text classification?" *CoRR*, vol. abs/1905.05583, 2019. [Online]. Available: http://arxiv.org/abs/1905.05583

Appendix A
Additional Tables

TABLE IV
AVERAGE ACCURACY ACROSS DIFFERENT BAG-OF-WORDS SETUPS

parameter	*value*	*score*
vectorizer	CountVectorizer	0.719
	TfidfVectorizer	0.727
stopwords	nltk	0.725
	none	0.703
	sk	0.74
lemmatize	0	0.723
	1	0.723
max features	100.0	0.361
	500.0	0.631
	1000.0	0.830
	2000.0	0.844
	3000.0	0.845
classifier	DecisionTreeClassifier	0.702
	GradientBoostingClassifier	0.729
	LogisticRegression	0.733
	MultinomialNB	0.704
	RandomForestClassifier	0.744
	SVC	0.723

TABLE V
BEST CNN MODEL BY MODE

	Metric		
embeddings	***mode***	***Accuracy***	***Loss***
fasttext	multichannel	0.928	6.75
fasttext	non-static	0.928	7.047
fasttext	static	0.908	8.805
own	multichannel	0.909	12.734
own	non-static	0.927	9.752
own	static	0.885	12.459
word2vec	multichannel	0.928	11.086
word2vec	non-static	0.927	6.668
word2vec	static	0.906	9.025

TABLE VI
HYPERPARAMETERS FOR THE BEST CNN MODELS

	Metric		
index	***fasttext***	***own***	***word2vec***
Test Accuracy	0.928	0.927	0.928
Dropout	0.8	0.8	0.8
Num. of Convolutions	250	250	250
Regularization	0.001	0.001	0.001