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
Faculty of Information Technology and Electrical Engineering
Univ. of Oulu
Oulu, Finland
anusha.ihalapathirana@student.oulu.fi

***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

Emotions are feelings that caused by a situation person interact with. It is also associate with person's character, mood, personality and motivation. People express their feelings and emotions using verbally and non-verbally such as words, intonation of voice, facial expressions, gestures, and tears. Nowadays people tend to communicate their ideas, opinions, and feelings through social medias, Such as tweeter, facebook, instergram. People filled with lot emotions and they commonly use written text to express their emotions and feelings through social medias. Categorizing text into emotion types is known as emotion analysis. Sentiment analysis detect positive, negative, and neutral feelings from text and emotion analysis detect the types of feelings, emotion state, in the text [1].

Emotion analysis takes major role in some applications which use emotion recognition, and it is a growing research area. There are supervised and unsupervised approaches can be found in this research area. A novel unsupervised context-based approach represents in [2] to detect emotions from text in sentence level. Their approached does not need any existing manually created lexicons and they used semantic relation between words and emotion type. They fine tune scores using syntactic dependencies within the sentence structure and proves this model provide more accurate results than other unsupervised approaches. Research carried out in [3] paid their attention towards capturing semantic features in the text. Authors used distributional semantic model to calculate the semantic relatedness of the emotion in the text and they achieved accuracy of 71.79% without training or annotation of data use.

A lot of research had been carried out on sentiment analysis and emotion analysis problem in many different languages. Bag-of-words representations, first suggested by Luhn in the 1950s [4], are an important advance, and still a commonly applied tool for natural language processing tasks like text classification. In these models, text is represented as a sparse feature matrix, where features represents the occurrences of a given word in a text. Earlier approaches involved counting the occurrences of words. Later, improvements have been suggested that offset the number of occurrences of a word in a given text by the frequency of the word's occurrence in other texts in the corpus [5]. The most notable of these approaches is the term frequency–inverse document frequency (tf-idf) weighting.

In a seminal, of applying machine learning algorithms using a bag-of-words representation [6] propose the use of support vector regressions with the tf-idf matrix approach. Using this approach, they achieve notable improvements over the state-of-the-art benchmarks at the time. In a more recent example, [7] demonstrate their work on supervised machine learning approach to recognize the emotion types using emotion annotated dataset which combined news headlines, fairy tales and blogs. They use bag of words and N-grams and proved that support vector machines classifier shows better performance and generalization than the other classifiers they used.

[8] introduce convolutional neural networkds for text categorization. Similar concepts are explored by e.g. [9], with a low complexity word level deep convolutional neural network for text categorization. [10] propose a new convolutional neural network that exploits from character to sentence level information to perform sentiment analysis of short texts. They found that combining small text with prior knowledge is effective.

Research in [11] introduce a new language representation model, BERT, a pretrained deep bidirectional representations from unlabeled text. They used conditioning on both information before and after a given word in all layers.BERT and successors inspired by it (see e.g. [12]), have massively improved the state-of-the-art in many language understandings tasks. For text classification, [13] used BERT provide general solution for BERT fine tuning in a text classification setting. Similarly, research on [14] improves the fine tuning of BERT using two effective mechanisms: self-ensemble and self-distillation.

In this research we are focusing on emotion analysis using word categorizations, semantic similarity approaches and machine learning approaches.

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 [15]. 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 [16]. 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 [17] and pre-trained fastText embeddings from [18]. 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.

B. Harvard Inquirer and String Matching

The first method we explore is to categorize each word in Harvard General Inquirer [19] to each emotion type. We use a filtering technique to this categorization. To identify words related to love emotion, we include Harvard General Inquirer words with that are labeled Affil label and exclude words associated with Negative label. The categories used for all six labels is reported in Table IX. We store this information into a database and then compute the occurrences of words associated with each label for each sentence and predict the label of a sentence to be the category with the highest number of words.

C. Empath Client Categories

The next method we explore is to use the Empath Client [20], a text analyzing tool with a pre-buildt category set. We identify categories related to each sentence using the Empath client. Then for all the identified categories, we compute empath category similarity with each label using the concept of lowest common hypernymy. We then associate each category provided by the Empath client with the emotion that has the highest similarity. Then, we use the label assigned to each category of a sentence to identify the label that has the highest number of occurrences. The emotion is predicted to be this label. The empath client categorization has a category for each of our labels as well, so we also experiment using only the exact emotion category.

D. Semantic Similarity

asd

E. SentiStrength Sentiment Scores

As part of the project specification, we also use the SentiStrength client [21], 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. However, intuitively, the more difficult task is distinguishing between different positive emotions and between different negative emotions. For this purpose, the sentiment scores are unlikely to provide added benefit. Considering this, and the fact that the simple approaches perform quite poorly, we do not explore the effects of these sentiment scores onto the accuracy of different models. Sentiment scores are still delivered as part of the project database, as suggested in the project specification.

F. Bag-of-Words Models

G. Convolutional Neural Networks

1) Model architecture: We replicate the convolutional neural network architecture used in [8]. 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 zero-padded to length n. [1] Then a set of 1-dimensional convolutions

[1] A similar architecture is proposed in [9], but one where sentences are not padded to a fixed length. Note that, the application of max-pooling is likely to make the zero-padding inconsequential.

are applied on the sentences. For each sentence, a feature c_i is generated by the convolution

$$c_i = f(w * x_{i:i+h-1} + b) \tag{1}$$

where h refers to the length of the convolution window, w are the filter weights of the convolution operation, b is an added bias term, and f is a non-linear activation function. A feature map c is then an $n-h+1$-dimensional vector of these features.

Equation (1) describes only one feature map produced by one convolution filter, while the model can consist of an arbitrary number of these feature maps. Then, for each feature map, max-pooling is applied, such that the largest feature value c_{i-max} is selected. This operation produces a vector of final features with dimension equal to the number of feature maps used. These features are then connected to the final, 6-dimensional softmax output with a fully connected layer.

For regularization the original paper uses dropout on the penultimate layer. Dropout is a commonly used regularization method in neural networks. It hides each of the final features from the output layer with some probability p, preventing the co-adaptation of the different features. In addition, we experiment with another common method of avoiding overfitting; an l_2-norm regularization across the layer weights. The l_2 regularization penalizes large weight vectors, preventing overfitting. Finally, following [8], we also constrain the l_2 -norms of the weight vectors w to be a maximum of a fixed size s.

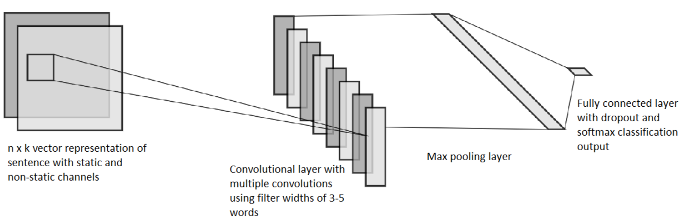

Fig. 1. CNN model architecture

In [8], pre-trained Word2Vec embeddings from [17] [18] are used. In addition to replicating this approach, we consider a few alternatives. First, we test pretrained fastText embeddings . Second, we explore explore training our own word2vec and fastText embeddings. The word2vec model... The fasttext model... For both models, we use a maximum distance between predicted words of five, and use early stopping on a word analogy evaluation task.

2) Hyperparameters and training: For all CNN models, we use sentence length $n = 70$ and convolution windows h of 3, 4 and 5. following the original paper. We use a rectified linear unit (ReLU) activation function for the convolutions, and p.

The original paper uses a set of 100 convolutions. We treat the number of convolutions as a hypermeter, and experiment with 50, 100, 150, and 300 convolutions. Note that the number of convolutions is applied for each of the three window lengths resulting in a total of $3*num.featuremaps$ convolutions. As for the dimension of the word embeddings, 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.

The original paper uses Adadelta [22] optimization, but we achieve faster convergence with an Adam [23] update rule. We train using mini-batches of 50 sentences, a learning rate of 10^{-3} and the cross-entropy loss function. We also employ early stopping based on the models cross-entropy loss on the test dataset.

IV. Results and Discussion

A. Word categorizations and semantic similarity

TABLE I
WORD CATEGORIZATIONS AND SEMANTIC SIMILARITY ACCURACIES

Classifier	**Accuracy**
Harvard String Matching	0.336
Empath Client	0.160
Empath Client with exact categories	0.183
Semantic Similarity - One synset	0.269
Semantic Similarity - All synset	0.240

B. Model selection for bag-of-words

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

TABLE II
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

C. Validation set performance of best models

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

TABLE III
VALIDATION AND TEST SET ACCURACIES FOR BEST MODELS

Classifier	Accuracy	
	Validation	*Test*
MultinomialNB	0.681	0.694
LogisticRegression	0.872	0.864
RandomForestClassifier	0.899	0.89
CNN fastText pretrained	0.925	0.928
CNN Word2Vec own	0.928	0.929
CNN fastText own	0.928	0.931
CNN Word2Vec pretrained	0.930	0.928

TABLE IV
VALIDATION PRECISION AND RECALL FOR BEST MODELS

Classifier	Metric	
	Precision	*Recall*
MultinomialNB	0.734	0.681
LogisticRegression	0.875	0.872
RandomForestClassifier	0.898	0.899
CNN fastText pretrained	0.926	0.925
CNN Word2Vec own	0.93	0.928
CNN fastText own	0.928	0.928
CNN word2vec pretrained	0.930	0.930

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, [11], [12] and [13] 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. [2]

Our results indicate that the starting point vectors are not of massive importance in tuning to the tasks, as different embedding representations achieve very similar test and validation set performance. Further exploring alternative setups for pre-training the word vectors might nonetheless be an interesting exercise but is beyond the scope of this paper. An important aspect of this observation, is the fact that 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.

In terms of the generalizability of the results, 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. [3]

...

VI. CONCLUSION

conclude.

VII. LATEX EXAMPLES

To be removed.

A. Equations example

$$a + b = \gamma \tag{2}$$

Use "(2)", not "Eq. (2)" or "equation (2)", except at the beginning of a sentence: "Equation (2) is . . ."

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] Chew-Yean, "Emotion detection and recognition from text using deep learning," 2015. [Online]. Available: https://devblogs.microsoft.com/cse/2015/11/29/emotion-detection-and-recognition-from-text-using-deep-learning

[2] A. Agrawal and A. An, "Unsupervised emotion detection from text using semantic and syntactic relations," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, 2012, pp. 346–353.

[3] R. Jan and A. A. Khan, "Emotion mining using semantic similarity," in *Natural Language Processing: Concepts, Methodologies, Tools, and Applications*. IGI Global, 2020, pp. 1115–1138.

[4] H. P. Luhn, "A statistical approach to mechanized encoding and searching of literary information," *IBM J. Res. Dev.*, vol. 1, no. 4, p. 309–317, Oct. 1957. [Online]. Available: https://doi.org/10.1147/rd.14.0309

[5] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513 – 523, 1988. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0306457388900210

[2] There is a minimal, unofficial implementation of BERT [11] embeddings on the dataset we study [15] available at Kaggle by the dataset author. This implementation appears to slightly exceed the validation accuracy of the best CNN model studied here by around 0.3 percentage units.

[3] While we do not have an out-of-distribution labeled dataset available, unreported, hand-crafted tests support this intuition.

[6] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Machine Learning: ECML-98*, C. Nédellec and C. Rouveirol, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 137–142.

[7] S. Chaffar and D. Inkpen, "Using a heterogeneous dataset for emotion analysis in text," in *Advances in Artificial Intelligence*, C. Butz and P. Lingras, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 62–67.

[8] 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

[9] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 562–570. [Online]. Available: https://www.aclweb.org/anthology/P17-1052

[10] C. dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin, Ireland: Dublin City University and Association for Computational Linguistics, Aug. 2014, pp. 69–78. [Online]. Available: https://www.aclweb.org/anthology/C14-1008

[11] 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

[12] 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

[13] 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

[14] Y. Xu, X. Qiu, L. Zhou, and X. Huang, "Improving bert fine-tuning via self-ensemble and self-distillation," *arXiv preprint arXiv:2002.10345*, 2020.

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

[16] 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

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

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

[19] "Harvard general inquirer," 2020. [Online]. Available: http://www.wjh.harvard.edu/ inquirer/homecat.htm

[20] "Empath client," 2020. [Online]. Available: https://github.com/Ejhfast/empath-client

[21] 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

[22] M. D. Zeiler, "ADADELTA: an adaptive learning rate method," *CoRR*, vol. abs/1212.5701, 2012. [Online]. Available: http://arxiv.org/abs/1212.5701

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

APPENDIX A
ADDITIONAL TABLES

TABLE V
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 VI
BEST CNN MODEL BY MODE

	Metric		
name	***mode***	***Accuracy***	***Loss***
CNN Word2Vec own	multichannel	0.927	6.404
	non-static	0.929	6.44
	static	0.900	10.404
CNN fastText own	multichannel	0.921	9.463
	non-static	0.931	6.599
	static	0.770	25.254
CNN fastText pretrained	multichannel	0.928	6.326
	non-static	0.928	6.133
	static	0.908	8.786
CNN word2vec pretrained	multichannel	0.928	6.583
	non-static	0.927	6.668
	static	0.906	9.025

TABLE VII
HYPERPARAMETERS FOR THE BEST CNN MODELS

	Metric		
Hyperparameter	***CNN fastText pretrained***	***CNN Word2Vec own***	***CNN fastText***
dev acc	0.928	0.929	0.931
dropout	0.5	0.75	0.75
num feature maps	100.0	150.0	100.0
regularization	0.0	0.001	0.001
word dim	300.0	50.0	50.0

TABLE VIII
CONFUSION MATRICES

Classifier	Label					
	joy	*sadness*	*anger*	*fear*	*love*	*surprise*
CNN Word2Vec own	664 32 40 1264	525 26 25 1424	254 16 21 1709	196 34 16 1754	158 30 20 1792	60 5 21 1914
CNN fastText own	674 38 30 1258	531 32 19 1418	258 25 17 1700	171 12 41 1776	155 22 23 1800	67 15 14 1904
CNN fastText pretrained	664 36 40 1260	520 17 30 1433	259 24 16 1701	195 35 17 1753	148 27 30 1795	64 11 17 1908
CNN word2vec pretrained	673 36 31 1260	524 29 26 1421	255 15 20 1710	189 27 23 1761	156 23 22 1799	64 9 17 1910
LogisticRegression	672 119 32 1177	516 73 34 1377	229 23 46 1702	155 23 57 1765	122 10 56 1812	51 7 30 1912
MultinomialNB	683 364 21 932	514 268 36 1182	93 3 182 1722	60 3 152 1785	12 0 166 1822	0 0 81 1919
RandomForestClassifier	659 72 45 1224	508 43 42 1407	241 21 34 1704	185 36 27 1752	142 21 36 1801	62 10 19 1909

TABLE IX
HARVARD GENERAL INQUIRER CATEGORY SELECTION

Label	Inclusion	
	Included	*Excluded*
surprise	Arousal	
joy	Positiv	Affil
love	Affil	Negativ
anger	Hostile	
sadness	Negativ	Hostile
fear	Weak	