DUTH at SemEval-2021 Task 7: Is Conventional Machine Learning for Humorous and Offensive Tasks enough in 2021?

Alexandros Karasakalidis **Dimitrios Effrosynidis** **Avi Arampatzis**

Database & Information Retrieval research unit,
Department of Electrical & Computer Engineering,
Democritus University of Thrace, Xanthi 67100, Greece
{alexkara23,deffrosy,avi}@ee.duth.gr

Abstract

This paper describes the approach that was developed for SemEval 2021 Task 7 Hahackathon: Incorporating Demographic Factors into Shared Humor Tasks (Meaney et al., 2021) by the DUTH Team. We used and compared a variety of preprocessing techniques, vectorization methods, and numerous conventional machine learning algorithms, in order to construct classification and regression models for the given tasks. We used majority voting to combine the models' outputs with small Neural Networks (NN) for classification tasks and their mean for regression for improving our system's performance. While these methods proved weaker than modern, deep learning models, they are still relevant in research tasks because of their low requirements on computational power and faster training.

1 Introduction

The underpinnings of humor have proven far more vexing than those of other emotional experiences. It is a highly subjective topic that various scholars have attempted to construct theories for understanding its fundamental elements in the studies of philosophy, linguistics, psychology and sociology. Some theories, e.g. the *Benign Violation Theory* (Warren and McGraw, 2015), suggest that humor can be described as linguistic violations that still make grammatical sense. The aforementioned theory supports that for a joke to be classified as humorous, it needs to avoid being too harmless or too offensive.

There are numerous studies in humor sentiment analysis in the last decade. In microblogging, Reyes et al. (2012) considered extracting linguistic devices from tweets to be used as features for classifying these tweets as humorous or ironic, while Raz (2012) approached the classification of humorous tweets as a multi-class problem of 11 types of humor, in his attempts to better attribute the real sentiment of a tweet. Recent attempts on humor detection on SemEval's 2020 Task 7 indicate that transformer models like BERT (Mahurkar and Patil, 2020) far outperform traditional machine learning algorithms (S et al., 2020).

This paper describes our submissions to SemEval 2021 task 7 and is structured as follows. Section 2 describes the tasks, training data, and evaluation measures. Section 3 describes key methods and algorithms used. Section 4 describes our proposed system while Section 5 analyzes our results. Finally, we draw our conclusions in Section 6, where we also propose directions for future work.

2 Background

In this section we describe each subtask's objective, the given data, and evaluation measures.

2.1 Subtasks

The main objectives of SemEval 2021 Task 7 were split into 4 subtasks. Subtask 1a required us to classify short texts as humorous or not, while Subtasks 1b and 1c required us to rate the text's humor and further classify it as controversial or not respectively. Finally, in Subtask 2, we had to rate how offensive each text was—humorus or not. All texts were in English.

2.2 Dataset

The organizers released the full training data in three parts: trial, development, and evaluation. Our final training dataset consisted of 9,000 different texts annotated with labels regarding each subtask in csv file format, while our test set consisted of 1,000 texts. Statistics for the training dataset are presented in Table 1 and Figure 1.

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 1125–1129
Bangkok, Thailand (online), August 5–6, 2021.

	Is humorous?	Is controversial?
Yes	5,564	2,773
No	3,436	2,791

Table 1: Humor and controversiality labels in the training set

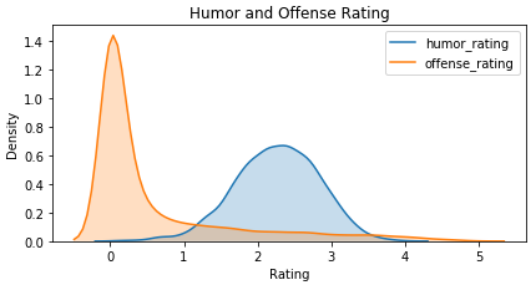

Figure 1: Humor and Offense Rating density plot

2.3 Evaluation Measures

For the classification Subtasks 1a and 1c, we use the F_1 measure. For the regression Subtasks 1b and 2, we use the root mean squared error (RMSE).

3 Experimental Setup

In this section we describe the preprocessing and vectorization methods as well as the machine learning algorithms used.

3.1 Preprocessing

Text preprocessing is the backbone of every text classification task. We applied the following techniques:

1. Tokenizing and Lowercasing words: We lower-cased the words in the texts and split them into tokens.

2. Stemming or Lemmatisation: Reducing noise from texts while (generally) improving system performance.

3. Removing Stopwords: Stopwords do not add much meaning to a sentence, so removing them helps in reducing the number of features and improving results.

4. Tagging words with capital letters: Tagging each word containing a capital letter that is not the first word in a sentence. Applied (when used) prior to word lowercasing.

5. Replacing Emojis: Very few emojis were found in texts, so we replaced them with their corresponding sentiment.

6. Replacing Contractions: We replace contractions into their full forms using a dictionary.

7. Removing integers: Numbers have no emotional value, so we remove them.

8. Part-of-Speech (POS) tagging: We used POS tagging of words for preprocessing the texts, following two different approaches.

   (a) Appending POS tags in words: Aims to incorporate part-of-speech information in our features.

   (b) Removing words based on their POS tag: Aims to remove words with low sentimental value from the data by targeting specific tags.

9. Numeric Feature Extraction: We extracted counts of characters, words, exclamation points, and numbers, as well as the numbers of declarative, interrogative, and imperative/exclamative sentences in a text. Finally, we extracted the counts of verbs, nouns, and adjectives, for each text.

The performance comparison of each preprocessing method is shown in Section 5. Stemming, lemmatisation, POS tagging, and most of numeric feature extractions were achieved by using tools from the well established NLTK (Elhadad, 2010), while we were guided by the survey of Ravi and Ravi (2015) of the most commonly used techniques in text preprocessing for sentiment analysis and by our previous works (Effrosynidis et al., 2017; Symeonidis et al., 2018) on this subject.

3.2 Machine Learning

The training of our classification and regression models aimed to improve the evaluation measure used for the corresponding task. Many probabilistic, linear and tree-based algorithms were used, as well as small neural network architectures.

The algorithms/Neural Networks (NN) that performed the best were used in our systems and are listed below:

- Linear Models: Linear SVM, Bayesian Ridge Regression and LASSO

- Non-Linear Models: Naive Bayes, Light GBM (Ke et al., 2017) and XGBoost (Chen and Guestrin, 2016)

- NN Models: Dense and Long Short-Term Memory Networks (using the Keras API[1])

Linear SVM models were excluded from our final systems mainly due to our better tuning of the LGBM and XGB models during the last phase.

3.3 Vectorization and Embedding

We used the following scikit-learn toolkit's vectorizers to extract features from the preprocessed data using word unigrams or bigrams:

- Tf-idf Vectorizer: translates the word counts matrix to a matrix of tf-idf features.
- Delta tf-idf Vectorizer: proposed by Martineau and Finin (2009), it creates tf-idf features similarly to tf-idf vectorizer but applies a weighting scheme reflecting the difference of tf-idf value of each word between the texts of two classes. We used the subtask's 1a labels for weighting tf-idf values in every subtask.

In order to create features for the LSTM models, we translate the words of each text into word vectors. This translation is achieved through the use of a well-known, pre-trained model: GloVe (Pennington et al., 2014).

4 System Overview

In this section we describe the proposed system for each subtask.

4.1 Proposed System

We trained each model with all possible combinations of preprocessing and vectorization. During the development phase, we evaluated these models using 10-fold cross validation on the training data. These evaluations guided us through hyperparameter tuning and model selection.

During the evaluation phase, we combined the outputs of these models in order to produce our system's predictions (Figure 2). The system's performance was evaluated using the test data from the development phase. That data was also used as validation data for training and tuning the dense and LSTM networks.

Model selection for our final systems was a repetitive but simple process. We selected the best performing model per algorithm and vectorization method, some weaker models whose outputs had lower correlation than the outputs of the best performing models and all the NN models that we had trained. We then combined (Symeonidis et al., 2017) all selected model outputs in all possible combinations consisting of at least 3 models and picked the best performing one as our final system for task prediction.

This process was repeated for each subtask. After finding the combination that produced the best results for the development's phase test data, we re-trained the system's models by appending that test data on the training data.

[1]https://github.com/fchollet/keras

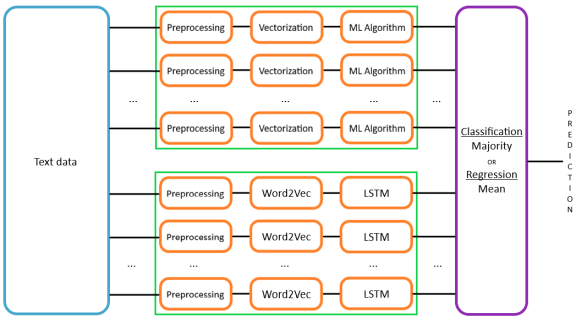

Figure 2: System Architecture

4.1.1 Subtask 1a: Humour Classification

In Subtask 1a, each text needs to be classified as humorous or not. Table 2 showcases the various preprocessing methods, vectorization tools and ML algorithms that comprised our final system. Since the number of models is even, majority voting favors the 'non-humorous' label in case of a tie, i.e. the less represented tag in the dataset.

Preprocessing	Vectorizer	ML
1,2,5,8a and 9	Delta tf-idf unigrams	Shallow NN
1,2 and 9	Tf-idf unigrams	LGBM
1,2,8a and 9	Delta tf-idf bigrams	Naive Bayes
1,2,6,7,8a and 9	Delta tf-idf bigrams	Naive Bayes
1,7 and 9	Tf-idf bigrams	XGB
1	Word2vec embedings	LSTM

Table 2: Subtask 1a system composition

4.1.2 Subtask 1b: Humor Rating

In Subtask 1b, each humorous text needs to be rated in a range of 0–5 on how much humorous it is. For this subtask, the best combination found amounted to 13 models. Thus, we will not include a table for this task. All proposed preprocessing techniques were used but 3 and 8b as well as every vectorizer. Interestingly, NN models were ruled out in this

subtask since a LGBM, XGB and Bayesian Ridge combination produced the best outcome.

4.1.3 Subtask 1c: Controversial Humor Classification

In Subtask 1c, each humorous text has to be classified as controversial or not. This is the only task that a single model outperformed any combination of models we tried to assemble. A preprocess of extracting numeric features (9), appending POS tags (8a), lowercasing and tokenization (1), a delta tf-idf vectorizer extracting bigrams and an LGBM model outperformed every other combination.

4.1.4 Subtask 2: Offense Rating

Finally, in Subtask 2 each text is rated in the range of 0–5 on how offensive it is. The final system is the average of the 7 individual models described in Table 3.

Preprocessing	Vectorizer	ML
1,2 and 9	delta tf-idf unigrams	XGB
1,2,3 and 9	tf-idf biwords	XGB
1	word2vec embedings	LSTM
1 and 2	word2vec embedings	LSTM
1 and 3	word2vec embedings	LSTM
1 and 5	word2vec embedings	LSTM
1 and 7	word2vec embedings	LSTM

Table 3: Subask 2 system composition

5 Results

Each preprocessing method had an impact on model performance, as it is shown in Table 4 for each task.

The results for each task/subtask are shown in Table 5. We present the scores of our best performing single models and combination of models respectively on the development test set as well as our submissions for the evaluation test data.

We can detect a pattern in the difference between the winning team's submissions and our submissions, and between the performance of our NN and non-NN models. It would be an indication—for Subtask 1a—that our conventional machine learning models, while achieving a respectable performance, cannot handle some outliers. This can be also observed through the results of Subtask 2, where outliers have a greater impact on the RMSE metric. Our false negative results on humor are mostly ironic, reference-based jokes or highly controversial ones, and our false positive

Preprocessing	Subtask 1a	Subtask 1b	Subtask 1c	Subtask 2
None	0.8325	0.5501	0.6299	0.8431
Stemming	**0.8372**	**0.5500**	0.6266	0.8560
Lemmatization	**0.8361**	0.5535	0.6130	0.8504
Stopwords	0.8206	**0.5465**	**0.6379**	0.8623
Capital Tagging	0.8284	**0.5550**	0.6099	0.8790
Emoji Replace	**0.8332**	0.5501	0.6299	0.8492
Contraction Replace	**0.8345**	**0.5471**	0.6253	0.8587
Integers Remove	**0.8357**	0.5567	0.6176	0.8664
POS Tag Append	**0.8331**	**0.5459**	**0.6455**	0.8620
POS Tag Filter	0.8170	0.5558	0.6214	1.0022
Feature Creation	**0.8457**	**0.5473**	0.5298	0.8653

Table 4: F_1 or RMSE of best performing model per preprocessing method. 'None' stands for plain tokenization and lowercasing. Scores in **bold** are better than 'None'.

results are mostly conversational writing texts like microbloging posts. The basic LSTM models we created were able to slightly close the gap with the superior, transformer models but there is still plenty of headroom for improvement.

On the other hand, our systems on Subtasks 1b and 1c were much closer to their superior models. While Subtask's 1b results could be attributed to a large extend on the distribution of humor ratings, Subtask 1c seems to be a much harder task regardless the approach. With an average accuracy of 0.5 across all submissions, humor controversiality seems to puzzle even the most complex models; anywise, humor controversiality is much more subjective than humor itself.

Nevertheless, a great advantage of conventional machine learning is training speed and hardware requirements. State-of-the-art boosting models like the ones we used (LightGBM and XGB) can be accelerated through the use of GPUs while our small NNs can be trained in a couple of minutes. Training/tuning deep learning models, on the other hand, requires expensive hardware and can be very time-consuming.

6 Conclusions

In this report, we presented our approach on humorous and offensive text classification and rating based on the combination of outputs from different preprocessing techniques, vectorization methods and machine learning algorithms. Our proposed systems were outperformed by other teams in the main tasks, while our conventional machine learning models were mostly inferior to our neural networks. Our future work will focus on expanding our preprocessing methods, introducing further ensemble methods and stacking, as well as

	Development Test			Evaluation Test		
	Best Single non-NN Model	Best Single NN Model	Best System	Best Submission	Our Position	Winning Submission
Subtask 1a	0.8706 (F_1)	0.8952 (F_1)	0.9136 (F_1)	0.8942 (F_1)	52 out of 58	0.9820 (F_1)
Subtask 1b	0.5411 (RMSE)	0.5492 (RMSE)	0.5411 (RMSE)	0.5507 (RMSE)	12 out of 50	0.4959 (RMSE)
Subtask 1c	0.6455 (F_1)	0.6491(F_1)	0.6636 (F_1)	0.5990 (F_1)	12 out of 36	0.6302 (F_1)
Subtask 2	0.8313 (RMSE)	0.7552 (RMSE)	0.7368 (RMSE)	0.5819 (RMSE)	40 out of 48	0.4120 (RMSE)

Table 5: Model/System performance in development and evaluation test sets

transformer-based models for direct comparisons.

References

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM.

Dimitrios Effrosynidis, Symeon Symeonidis, and Avi Arampatzis. 2017. A comparison of pre-processing techniques for twitter sentiment analysis. In *Research and Advanced Technology for Digital Libraries - 21st International Conference on Theory and Practice of Digital Libraries, TPDL 2017, Thessaloniki, Greece, September 18-21, 2017, Proceedings*, volume 10450 of *Lecture Notes in Computer Science*, pages 394–406. Springer.

Michael Elhadad. 2010. Natural language processing with python steven bird, ewan klein, and edward loper (university of melbourne, university of edinburgh, and BBN technologies) sebastopol, CA: o'reilly media, 2009, xx+482 pp; paperbound, ISBN 978-0-596-51649-9, $44.99; on-line free of charge at nltk.org/book. *Comput. Linguistics*, 36(4):767–771.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3146–3154.

Siddhant Mahurkar and Rajaswa Patil. 2020. LRG at semeval-2020 task 7: Assessing the ability of BERT and derivative models to perform short-edits based humor grading. *CoRR*, abs/2006.00607.

Justin Martineau and Tim Finin. 2009. Delta TFIDF: an improved feature space for sentiment analysis. In *Proceedings of the Third International Conference on Weblogs and Social Media, ICWSM 2009, San Jose, California, USA, May 17-20, 2009*. The AAAI Press.

J.A. Meaney, Steven R. Wilson, Luis Chiruzzo, Adam Lopez, and Walid Magdy. 2021. Semeval 2021 task 7, hahackathon, detecting and rating humor and offense. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

Kumar Ravi and Vadlamani Ravi. 2015. A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. *Knowl. Based Syst.*, 89:14–46.

Yishay Raz. 2012. Automatic humor classification on twitter. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 3-8, 2012, Montréal, Canada*, pages 66–70. The Association for Computational Linguistics.

Antonio Reyes, Paolo Rosso, and Davide Buscaldi. 2012. From humor recognition to irony detection: The figurative language of social media. *Data Knowl. Eng.*, 74:1–12.

Kayalvizhi S, Thenmozhi D., and Aravindan Chandrabose. 2020. SSN_NLP at SemEval-2020 task 7: Detecting funniness level using traditional learning with sentence embeddings. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 865–870, Barcelona (online). International Committee for Computational Linguistics.

Symeon Symeonidis, Dimitrios Effrosynidis, and Avi Arampatzis. 2018. A comparative evaluation of pre-processing techniques and their interactions for twitter sentiment analysis. *Expert Syst. Appl.*, 110:298–310.

Symeon Symeonidis, Dimitrios Effrosynidis, John Kordonis, and Avi Arampatzis. 2017. DUTH at semeval-2017 task 4: A voting classification approach for twitter sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 704–708. Association for Computational Linguistics.

Caleb Warren and A. Peter McGraw. 2015. Benign violation theory. *Mays Business School*, 2015-11.