CLaC @ DEFT 2018: Sentiment analysis of tweets on transport from Île-de-France

Simon Jacques   Farhood Farahnak   Leila Kosseim
Dept. of Computer Science and Software Engineering
Concordia University
1455 De Maisonneuve Blvd. W. Montreal, Canada
s-jacques@live.com, farhood.farahnak@gmail.com,
leila.kosseim@concordia.ca

RÉSUMÉ

Analyse de tweets sur les transport sur l'Île-de-France

Cet article décrit le system développé par le laboratoire CLaC de l'Université Concordia à Montréal pour la campagne DEFT 2018. La compétition comptait quatre tâches differentes, parmis lesquelles nous avons participé aux deux premières. Nous avons utilisé deux méthodes d'apprentissage supervisé: une machine à vecteurs de support et un réseau de neuronnes. À la tâche 1, notre mesure-F la plus élevée atteint 87.61% et à la tâche 2, elle atteint 51.03%, situant notre système en dessous de la moyenne par rapport aux autres participants.

ABSTRACT

CLaC @ DEFT 2018: Analysis of tweets on transport on the Île-de-France

This paper describes the system deployed by the CLaC lab at Concordia University in Montreal for the DEFT 2018 shared task. The competition consisted in four different tasks; however, due to lack of time, we only participated in the first two. We participated with a system based on conventional supervised learning methods: a support vector machine classifier and an artificial neural network. For task 1, our best approach achieved an F-measure of 87.61%; while at task 2, we achieve 51.03%, situating our system below the average of the other participants.

MOTS-CLÉS : Machine à vecteurs de support; Analyse de sujets; Anlyse de sentiments.

KEYWORDS: Support Vector Machine; Topic Analysis; Sentiment Analysis.

1 Introduction

This paper describes the system deployed by the CLaC Lab at Concordia University for the DEFT 2018 shared task. For this 14^{th} edition of the *Défi Fouille de Textes* (DEFT), the main goal was to analyze the sentiments in French tweets regarding transport on the Île-de-France. As described in (Paroubek et al., 2018), four tasks were proposed:

task 1 (T1) – Transport / non-transport classification: Given a tweet, determine whether it concerns transport or not.

task 2 (T2) – Global polarity: Given a transport tweet, determine its overall polarity, chosen from 4 potential classes: positive, negative, neutral or mixed (mixposneg).

Task 3 (T3) – Sentiment marker and target: Given a tweet about transport that expresses at least one sentiment, for each sentiment expressed, determine (1) the target of the sentiment, and (2) the sentiment marker (i.e. the linguistic expression that expresses the sentiment).

Task 4 (T4) – Full annotation: Given a tweet about transport that expresses at least one sentiment, for each sentiment expressed, determine (1) the target of the sentiment, (2) the sentiment marker, in addition to (3) the source of the sentiment.

As each task builds on top of the previous one, due to lack of time, we only participated in the first two tasks (T1 and T2).

2 Datasets

The DEFT 2018 organizers (Paroubek et al., 2018) put at our disposal a training corpus of 68,916 tweets already annotated with their topic label (*transport/non-transport*) and their polarity label (*positive, negative, neutral, mixed*). To train our classifiers, we first split the dataset randomly to create two distinct sub-sets: (i) 80% was used as training set, (ii) and 20% was used as validation set. Since the polarity was identified only on transport tweets, the second task consisted of 28,374 tweets for training and 7,094 for validation. Tables 1 and 2 show the distribution of the datasets for tasks T1 and T2, respectively.

Table 1: Training and validation sets for task 1

Label	**Training**	**Validation**	**Total**	**Proportion**
Transport	28,374	7,093	35,468	51.47%
Non-Transport	26,758	6,690	33,448	48.53%
Total	55,132	13,783	68,916	100%

Table 2: Training and validation sets for task 2

Label	**Training**	**Validation**	**Total**	**Proportion**
Neutral	10,089	2,522	12,611	35.56%
Positive	5,862	1,466	7,328	20.66%
Negative	10,487	2,622	13,109	36.96%
Mixposneg	1,936	484	2,420	6.82%
Total	28,374	7,093	35,468	100%

3 Pre-Processing

Given the original datasets (see Section 2), we first performed various pre-processing steps to clean and normalize the tweets. These steps were inspired by the work of (Mohammad et al., 2013; Pak & Paroubek, 2010; Reitan et al., 2015).

1. **UTF-8 Encoding**: To facilitate processing, all messages in the corpus were encoded with the format *UTF-8*.

Step	Message
1. UTF-8 Encoding	"#EURO2016 #ALLFRA : à #Lille , même les bus supportent les #Bleus. C’ est le Noooord https://t.co/ws5s9LyiR7 https://t.co/HEX9ksiPSH"
2. Hypertext Removal	"#EURO2016 #ALLFRA : à #Lille , même les bus supportent les #Bleus. C’ est le Noooord"
3. Case folding	"#euro2016 #allfra : à #lille , même les bus supportent les #bleus. c’ est le noooord"
4. Special Character Removal	"euro2016 allfra à lille même les bus supportent les bleus. c’ est le noooord"
5. Character Repetition Reduction	"euro2016 allfra à lille même les bus supportent les bleus. c’ est le noord"
6. Tokenization	euro2016, allfra, à, lille, même, les, bus, supportent, les, bleus, ., c’, est, le, noord
7. Stopword Removal	euro2016, allfra, lille, bus, supportent, bleus, ., noord

Figure 1: Example of pre-processing of tweets

2. **Hyperlink Removal**: All links starting with *http*, *https*, or *www* were removed from the tweets.

3. **Case Folding**: All uppercase characters were converted to lowercase.

4. **Special Character Removal**: To limit the character set, we only considered 45 possible characters. This 45-character set was composed of all 26 French letters, plus 12 with diacritics, and 7 punctuation marks including the hyphen. All characters not included in this predefined set, in particular hashtags (#), were removed from the tweets. This allowed us to focus the classification on words.

5. **Character Repetition Reduction**: All words that included more than two consecutive identical characters were reduced to only two consecutive characters. For example, as shown in Figure 1, *noooord* was reduced to *noord*.

6. **Punctuation Removal**: For the task 1, we removed all punctuation marks; however, as punctuation has been shown to signal sentiment (Mohammad et al., 2013), we kept them for task 2.

7. **Tokenization**: Once the characters were pre-processed, we tokenized the filtered tweets. For this, we used the French version of `word_tokenize` from the NLTK Toolkit (Loper & Bird, 2002).

8. **Stopword Removal**: We used a list of 156 stopwords to further filter the tweets. 130 stopwords came from the NLTK Tookkit (**?**), and the remaining 26 were added following a manual corpus analysis of the word distribution in the DEFT-2018 dataset.

Figure 1 illustrates the pre-processing of a sample tweet. As shown in Table 3, after pre-processing, the size of tweets was reduced to almost half their size for each label and for both tasks.

As part of the pre-processing, we also experimented with marking negation, in order to increase our performance on task T2. As shown by several previous work(e.g. (Reitan et al., 2015; Kouloumpis

Table 3: Average tweet size before and after pre-processing

		Average Nb Words	
Task	**Label**	**Before Pre-processing**	**After Pre-processing**
T1	Transport	22.13	12.17
	Non-Transport	22.62	12.59
T2	Neutral	21.36	11.76
	Positive	22.10	12.07
	Negative	22.58	12.47
	Mixposneg	23.82	13.10

et al., 2011), negation is an important feature for sentiment analysis on tweets. We therefore tried to mark the scope of negation by marking each expression indicating a negation (e.g. *pas, ne, n'*) until the next punctuation. Unfortunately, our elementary method to mark negation seemed to lower the performance with our baseline model, a Naive Bayes Classifier (see Section 4.2) on the the validation set. With the negation marking, the F-measure scored approximately 15% lower than without it. Hence, we dropped our negation marking in the pre-processsing.

4 Experiments

4.1 Features and Feature Selection

We experimented with two types of features and two feature selection methods for a total of 4 experiments. As features, we used (1) Words as feature with binary bag-of-words and frequency bag-of-words representation. (2) as character n-grams have successfully been used on tweets in previous work (e.g. (Reitan et al., 2015)), we also considered character n-grams as features with frequency. We experimented with 3-grams, 4-grams, and 5-grams. While the 5-gram model was too large to be trained efficiently, the 3-gram and 4-gram models were later dropped due to their low performance on the validation set.

As for feature selection, we experimented with 2 simple methods: (1) removing low-frequency features ($< s$ome value n), and (2) removing features with a low entropy difference ($\leq$ some value t) between the classes.

4.2 Models

We experimented with 4 different classifiers for task 1, and 3 classifiers for task 2.

1. **Naive Bayes Classifier**: As a baseline, we trained a Naive Bayes Classifier from the NLTK library. We used words as features and Boolean values indicating their presence or absence in the tweet as feature values. This lead to an F-measure of 79.91% on the validation set for task 1; but on task 2, however, the F-measure dropped to 64.18%, which we attempted to improve by using other models.

Table 4: Confusion matrix of the ANN for task 1 on the validation set

Predicted Actual	Transport	Non-Transport	Total
Transport	4340	203	4543
Non-Transport	1264	3060	4324
Total	5604	3263	8867

F-Measure	85.54%

2. **Decision Tree Classifier**: The second model we trained was a Decision Tree Classifier, also from the NLTK library, using the same features vectors as the Naive Bayes classifier. This model lead to an F-measure of 69.97% for task 1 and 57.20% for task 2 on the validation set. It scored much lower than the Naive Bayes classifier, which lead us to drop it entirely for the official runs.

3. **Support Vector Machine**: The third model we trained was a Support Vector Machine classifier from the *scikit-learn* library (Pedregosa et al., 2011). Four versions of this model were experimented with: two models using word-feature, and two models using character n-grams.

   For task 1, we used four different approaches to train our model. When we used words as features and filtered out words with frequency < 4 (i.e. $n = 4$, see Section 4.1), we reached an F-measure of 83.00% on the validation set. On the other hand, when filtering features with an entropy difference $= 0.25$ (see Section 4.1), we reached an F-measure of 83.05% on the validation set. We then trained with character 3-grams and 4-grams, but only reached F-measures of 69.14% and 78.16% respectively. Because the n-gram models achieved a lower performance than the word-based models, we did not use them for the actual shared task (see Section 5).

   For task 2, due to lack of time, we only experimented with two different approaches to train our model. The first one was based on words with frequency $= 4$ as features and an entropy difference $= 0.25$. This model reached an F-measure of 68.47% on the validation set. The second model used character 4-grams as features, since they seemed to achieve a higher performance than 3-grams on task 1. The 4-gram only managed to reach an F-measure of 42.72%, significantly lower than the word-based model. Again, the n-gram models were therefore discarded.

   Table 5 shows the confusion matrix of the best validation for this model on task 2.

4. **Artificial Neural Network**: The last model we trained was an classic Neural Network. We used words with frequency $= 4$ as features and an entropy difference $= 0.25$. Our model used a binary bag-of-words representation at the input layer, used two layers with the ReLU activation function (Nair & Hinton, 2010) and trained with the RMSprob optimization algorithm (Tieleman & Hinton, 2012). We also applied dropout (Srivastava et al., 2014) after each layer to prevent over-fitting. This model achieved an F-measure of 85.54% on the validation set.

Following the results of our experiments with the validation set, the best models seemed to be the ANN for task 1, and the SVM with the binary bag-of-words filtered with $n = 4$ and $t = 0.25$ for task 2. Tables 4 and shows the confusion matrix of the ANN model on the validation set. As indicated in Section 5, this model was used as a submission for task 1. Table 5 shows the confusion matrix of

Table 5: Confusion matrix of the SVM with $n = 4$ and $t = 0.25$ on the validation set

Predicted **Actual**	**Positive**	**Negative**	**Neutral**	**Mixed**	**Total**
Positive	838	128	102	173	1,241
Negative	174	1849	292	215	2,530
Neutral	378	633	2106	41	3,158
Mixed	56	39	5	64	164
Total	1,446	2,649	2,505	493	7,093

F-Measure	68.47%

the best performance of the SVM model for task 2 on the validation set. This configuration was used as a submission for Task 2 (see Section 5).

5 Results and Analysis

For the shared task, we submitted 2 runs for task 1 and 2 runs for task 2.

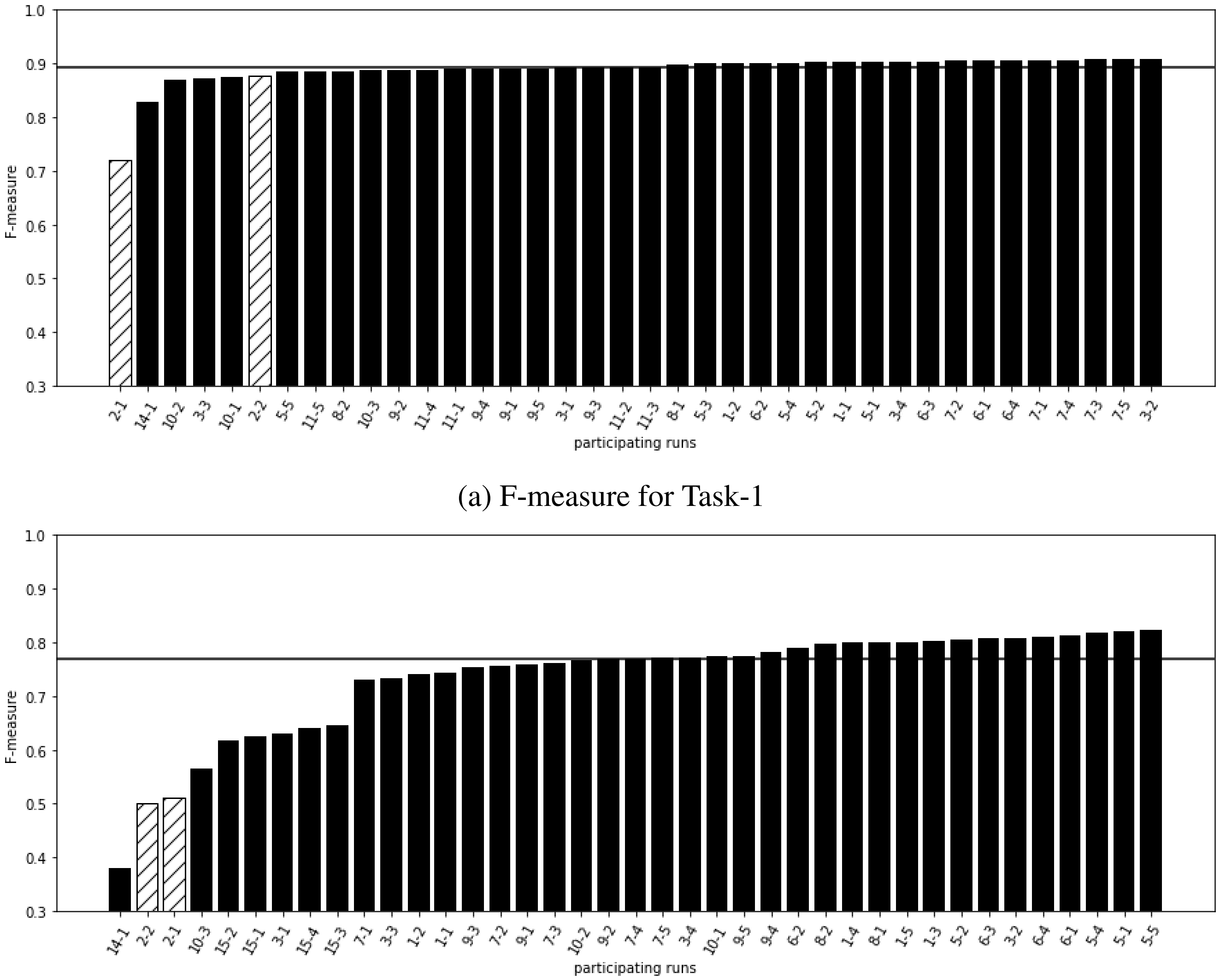
5.1 Runs for Task 1

We submitted 2 runs for task 1: CLaC_T1_run1 and CLaC_T1_run2.

CLaC_T1_run1: Consisted of the best SVM model described in Section 4.2, using the binary bag-of-words approach with word frequency cutoff $n = 4$ and entropy difference cutoff $t = 0.25$. While the performance with the validation set achieved an F-measure of 83.05%, our final result with the test set dropped to 71.90%. Table 6 shows the official results of the SVM run at Task 1.

CLaC_T1_run2: Consisted of the ANN model described in Section 4.2, also using the binary bag-of-words approach with word frequency cutoff $n = 4$ and entropy difference cutoff $t = 0.25$. Although the validation results for the ANN classifier showed an F-measure of 85.54%, our final F-measure with the test set increased to 87.61%. Table 6 shows the official results of the ANN run. Compared to the SVM classifier of CLaC_T1_run1, the ANN achieved better results at classifying *transport* messages, with a precision of 77.96% versus 56.14% for the SVM.

Figure 2 compares the F-measure of all participants for tasks 1 and 2, and indicates the median score by a horizontal line. As the figure shows, for task 1, the SVM (run 2-1 in the Figure) achieved the lowest f-measure among all participants; while the ANN (run 2-2) was close to the median of all participants. This difference in performance during the official runs was surprising, as these these two classifiers achieved very similar results with the validation set. However, as indicated in Table 6, the ANN out-performed the SVM by a difference of more than 16% in F-measure. We suspect that this large gap stems from the fact that, although we fine-tuned the hyper parameters of both models using the validation set, the dropout of the ANN reduced over-fitting. We also believe that the simple feature used in this task, contributed to their low performance.



(a) F-measure for Task-1

(b) F-measure for task-2

Figure 2: F-measure of all participants in task 1 (a) and task 2 (b)

	Run	
Measure	**CLaC_T1_run1**	**CLaC_T1_run2**
true positive	4387	6093
false_positive	3428	1723
false_negative	1	0
(micro-mean) precision	0.56136	0.77955
(micro-mean) recall	0.99977	1
(micro-mean) F1-measure	0.719	0.87612

Table 6: Official results for task 1

	Run	
Measure	**CLaC_T2_run1**	**CLaC_T2_run2**
true positive	1350	1320
false_positive	2591	2621
false_negative	0	0
(micro-mean) precision	0.34255	0.33494
(micro-mean) recall	1	1
(micro-mean) F1-measure	0.5103	0.50181

Table 7: Official results for task 2

5.2 Runs for Task 2

For task 2, we submitted two runs.

CLaC_T2_run1: Consisted of the SMV using the same binary bag-of-words approach with word frequency cutoff $n = 4$ and entropy difference cutoff $t = 0.25$. The validation results for the classifier showed an F-measure of 68.47%, while the final F-measure dropped to a low 51.03%.

CLaC_T2_run2: Consisted of a similar SVM, but trained on a more powerful machine, allowing us to use $n = 3$. In retrospect, increasing the number of features did not turn out to be a successful approach, as the final F-measure dropped to a low 50.18%, placing us again at the low end of the scores.

Figure 2b, shows that the first SVM (run 2-1) achieved the third lowest F-measure, and our second SVM (run 2-2) achieved the second lowest F-measure. These results were rather surprising as they were approximately 18% lower than those achieved during our validation runs. As with our runs at task 1, we suspect that the models over-fitted the training set due to the large number of features used. With the validation set, the F-measure for mixed tweets was only 19.48% for the SVM classifier, with $n = 4$ and entropy difference $t = 0.25$, much lower than the other three sentiment labels. This seems to show that the SVM classifier for sentiments achieved better results when classifying tweets with a single polarity than those with mixed polarity. As seen in Table 2, the proportion of mixed polarity messages in the training set was significantly lower than the other 3 sentiment labels; this might also have contributed to this low performance.

6 Conclusion

This paper described our first participation to the DEFT shared task. Due to lack of time, we only participated to the first two tasks: transport / non-transport classification and global polarity. We deployed models based on standard hand-crafted features and used off-the-shelf toolkits to pre-process the tweets and experiment with a variety of supervised learning models.

Although our results with the validation set seemed somewhat acceptable, most of our runs underperformed with the actual test set. Our results at the shared tasks clearly indicate that training with the Support Vector Machine classifiers seemed to over-fit the training set with the large feature set that we used; whereas the Artificial Neural Network seemed more robust as is reached 85.54% at task 1 with very little fine-tuning of hyper-parameters.

Acknowledgement

The authors would like to thank the organizers of the DEFT-2018 challenge. We gratefully acknowledge the financial support of NSERC and the NVIDIA Corporation for the donation of the Titan X Pascal GPU used for this research.

References

KOULOUMPIS E., WILSON T. & MOORE J. (2011). Twitter Sentiment Analysis: The Good the Bad and the OMG! In Proceedings of the International AAAI Conference on Web and Social Media, p. 538–541, Barcelona, Spain.

LOPER E. & BIRD S. (2002). NLTK: The Natural Language Toolkit. In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics (ETMTNLP-2002), p. 63–70, Philadelphia, USA.

MOHAMMAD S. M., KIRITCHENKO S. & ZHU X. (2013). NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In Proceedings of the Seventh International Workshop on Semantic Evaluation Exercises (SemEval-2013), p. 321–327, Atlanta, USA.

NAIR V. & HINTON G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-2010), p. 807–814, Haifa, Israel.

PAK A. & PAROUBEK P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. In Proceedings of the International Conference on Language Resources and Evaluation (LREC-2010), volume 10, p. 1320–1326, Valletta, Malta.

PAROUBEK P., GROUIN C., BELLOT P., CLAVEAU V., ESHKOL-TARAVELLA I., FRAISSE A., JACKIEWICZ A., KAROUI J., MONCEAUX L. & TORRES-MORENO J.-M. (2018). DEFT2018 : recherche d'information et analyse de sentiments dans des tweets concernant les transports en île de france. In Actes de DEFT, Rennes, France.

PEDREGOSA F., VAROQUAUX G., GRAMFORT A., MICHEL V., THIRION B., GRISEL O., BLONDEL M., PRETTENHOFER P., WEISS R., DUBOURG V., VANDERPLAS J., PASSOS A., COURNAPEAU D., BRUCHER M., PERROT M. & DUCHESNAY E. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, **12**, 2825–2830.

REITAN J., FARET J., GAMBÄCK B. & BUNGUM L. (2015). Negation scope detection for twitter sentiment analysis. In Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA), a workshop collocated with EMNLP, p. 99–108, Lisboa, Portugal.

SRIVASTAVA N., HINTON G., KRIZHEVSKY A., SUTSKEVER I. & SALAKHUTDINOV R. (2014). Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, **15**(1), 1929–1958.

TIELEMAN T. & HINTON G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning, **4**(2), 26–31.