Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference “Dialogue 2018”

Moscow, May 30—June 2, 2018

USING MACHINE TRANSLATION FOR AUTOMATIC GENRE CLASSIFICATION IN ARABIC

Bulygin M. V. (bulyginmv1996@gmail.com)[1],
Sharoff S. A. (s.sharoff@leeds.ac.uk)[1,2]

[1]Russian State University of Humanities, Moscow, Russia
[2]Leeds University, Leeds, UK

This paper addresses the task of automatic genre classification for Arabic within the Functional Text Dimensions framework, which allows texts to get a reliable genre description, while maintaining an adequate amount of genre labels. Our aim in this study is to build an automatic classification model that can annotate any Web text in Standard Arabic in terms of genres. To build the training corpus we translated English and Russian annotated texts into Arabic using Google MT. For building the model experimented with various machine learning approaches, such as Logistic Regression, SVM, LSTM, and different features, such as words, character n-grams and embedding vectors. For testing the classification models, we collected and annotated in terms of FTDs our own corpus of Arabic Web texts. The best performing model offers reasonable classification accuracy in spite of being based on a training corpus produced by MT.

Key words: Functional Text Dimensions, genre classification, machine translation, Web corpora annotation

ИСПОЛЬЗОВАНИЕ МАШИННОГО ПЕРЕВОДА В ЗАДАЧЕ АВТОМАТИЧЕСКОЙ ЖАНРОВОЙ КЛАССИФИКАЦИИ ДЛЯ АРАБСКОГО ЯЗЫКА

Булыгин М. В. (bulyginmv1996@gmail.com)[1],
Шаров С. А. (s.sharoff@leeds.ac.uk)[1,2]

[1]Российский государственный гуманитарный университет, Москва, Россия
[2]Университет Лидса, Лидс, Великобритания

1. Introduction

Understanding genre is essential for processing and comprehending different kinds of texts. One way to define a genre is as "a set of conventions that transcend individual texts, and create frames of recognition governing document production, recognition and use." [Santini et al., 2010] In our lives we encounter all sorts of texts, and our ability to classify them as different genres makes working with them easier. By grouping texts together, we can identify some regularities; this, in turn, helps us understand their communicative purpose and context. From this we can build our expectations of this text, our reaction to it and our response. Automatic genre classification using methods of computational linguistics is especially useful when we are talking about the overwhelming amount of texts from the web.

The Arabic language presents some issues for the researchers in the field of automatic genre classification. First, due to Islam playing an integral part in the Arabic community, we see many more texts about religion in Arabic data as opposed to data from other languages. In our research we are avoiding this disbalance when collecting texts for test annotation. We do this in order to get a well-rounded testing of our model. Another issue that a researcher working with Arabic faces is the dominance of Arabic dialects in informal communication. According to Ethnologue, there are 36 variations of the Arabic language [Ethnologue, 2015]. They differ from each other substantially, and thus cannot be treated as the same language. Standard Arabic is the language that unites all regions of the Arab world; however, no one acquires it as his or her mother language. Because Arab children typically learn Standard Arabic in schools (which not every child has the opportunity to attend [Vasil'ev, p.c.]), it is not spoken by the entire population. On social media Arabs tend to write in their regional dialects, because these constitute the language of communication with friends and family in daily life. For our model we used texts only in Standard Arabic.

There are many different systems of distinguishing between genres. Some of them tend to present long lists of genre labels that cover all possible varieties of texts. For example, [Görlach, 2004] classifies texts into 2100 different genres and [Adamzik, 1995] presents a list of over 4000 genres. These classifications are aimed at covering all of the different possibilities that exist in the world, but they are impractical for corpora purposes. Corpora require a smaller number of genre labels to collect reasonably sized subcorpora and to make it possible to compare language use between them [Sharoff, 2018]. Long lists are also not sensitive to genre hybridism. This is even more significant for Web texts, where boundaries between genres are not strict and people can blend them or violate certain conventions.

In this work we are using the FTD (Functional Text Dimension) approach for genre classification. It provides coverage power that is similar to that of long list genre systems, while maintaining an adequate amount of genre labels. It is also much more flexible and sensitive to genre hybridism. This is possible due to the description of a text's genre through a combination of several parameters at the same time. Thus, a text may receive not only typological, but also topological analysis [Sharoff, 2018]. The texts are distinguished among 18 Functional Text Dimensions, which represent a functional category instead of an atomic label, as in other genre classification systems. Whether a text belongs a given functional dimension is decided by a system of key questions. The texts can be scored on the following scale:

FTD value	The level of pertaining to the FTD
2.0	Strongly
1.0	Somewhat or partly
0.5	Slightly
0.0	None or hardly at all

With “0” or “None” being the default value for a FTD. The non-zero FTD values are assigned judging on how close the text is to a prototypical representative for this FTD. In our research we treated the “0.5” FTD value like the “0” score. That is why, the “0.5” score is not represented in evaluation.

2. Data

In this study we used a collection of annotated texts prepared by [Sharoff, 2018]. This collection consists of texts from 3 corpora: 5g—the Pentaglossal corpus [Forsyth and Sharoff, 2014], ukWac [Baroni et al., 2009] and GICR [Piperski et al., 2013]. 5g presents a set of texts coming from fiction, political debates, TED talks, etc. For our experiment we collected texts in English from the 5g corpus. ukWac consists of Web texts and news in English, which were collected by crawling the .uk domain. The GICR corpus represents a variety of genres such as news, articles from Wikipedia and several blogging platforms, collected from the Russian Web.

We used the method of Machine Translation to translate these annotated corpora to Standard Arabic. The resulting corpus was used as training data for our model (see **Table 1**)

Table 1: Size and composition of the training corpus

Corpus	Documents	Words
5g	247	686,568
ukWac	257	179,235
GICR	806	837,868
Total	**1,310**	**1,703,671**

We used Google Translate for translation, because it shows good performance and is accessible to any user. The quality of translation was quite high, because Google Translate managed to preserve most grammatical relations and syntactic boundaries. The most common mistake was inadequate word choice. For example, the Russian sentence “Спасибо, только день рождения был более полугода назад” (“Thank you, except the birthday was more than half a year ago”) was translated to the following:

(1)	shukran	eid	milad	faqat	kan	‘akthar	min	nisf	eam
	THANK YOU	HOLIDAY	OF BIRTH	ONLY	WAS	MORE	THAN	HALF	YEAR

However, we would expect to see this:

(2)	shukran	lakin	eid	milad	kan	‘akthar	min	nisf	eam
	THANK YOU	BUT/ONLY	HOLIDAY	OF BIRTH	WAS	MORE	THAN	HALF	YEAR

In our research we decided to conduct an experiment to find out whether a model can be trained on a corpus translated with Machine translation. The resulting program would be judged by how well it can classify natural Arabic texts.

For the testing set of data we collected and annotated 100 Arabic texts from the Web. There were 24 different sources: news, fiction, Wikipedia, scientific texts, law texts, etc. We did not use texts from Facebook or Twitter, because people do not use Standard Arabic there. The same issue takes place with forum dialogs and simple discussions. Text length varies between 300 and 1500 words. Each text was annotated in terms of Functional Text Dimensions, with all principal dimensions being scored. Statistics for the testing data set are presented in Table 2 with overall sum of annotations and mean value for each dimension.

Table 2: Distributions of genres in testing data set in terms of FTDs

FTDs	A1	A3	A4	A5	A6	A7	A8	A9	A11
Total	42	7	16	2	0	35	39	10	10
Mean	**0.42**	0.07	0.16	0,02	0	0.35	0.39	0.1	0.1
FTDs	**A12**	**A13**	**A14**	**A15**	**A16**	**A17**	**A18**	**A19**	**A20**
Total	5	0	10	1	45	15	0	0	0
Mean	0.05	0	0.1	0.01	**0.45**	0.15	0	0	0

3. Experiments

3.1. Features

In this study we conducted several experiments with different features and methods for classification. For the feature selection testing we chose words and character trigrams. Genre information is strongly concentrated around word choice. Stylistic differences are often key to genre identification. To vectorize word features we used tf-idf technic [Pedregosa et al. 2011], which is one of the most common methods for text vectorization. To vectorize texts for the LSTM neural network we used pre-trained word vectors from the fastText database [Bojanowski et al. 2016]. These vectors with a dimensionality of 300 were trained on the Arabic sector of Wikipedia, using the skip-gram model.

As mentioned in [Zhang et al. 2015], the character level of a text can be very useful for text classification. For our research we used character trigrams. Text tokenization was done using scikit-learn preprocessing tools. Trigrams were also vectorized using tf-idf technic.

3.2. Methods

For each of the features we used the following methods:

1. SVM
2. Logistic regression
3. XGboost
4. LSTM

The Support Vector Machine is one of the most popular methods for learning algorithms. It is used for various tasks of machine learning, such as sentiment analysis, language modeling and text classification. It is also useful for multiclass classification as shown in [Hou et al. 2015]. In our research we set the C value of the SVM model to 1, and we used the one-versus-rest scheme for the training of the classifier. The Support Vector Machine can also learn using different kernel types: linear kernel, RBF kernel and polynomial kernel. For our model we used linear kernel.

Logistic regression is a basic method of machine learning borrowed from field statistics. It utilizes logistic function to predict the probability of an answer. It is one of the most popular methods for binary classification, but one can also use it for multiclass classification, for example, in tasks of image classification or spam filtering. The C parameter of our Logistic regression model was set to 1. We also used the one-versus-rest scheme to train the classifier.

XGBoost (Extreme Gradient Boosting) is an implementation of gradient boosted decision trees designed to increase productivity and performance. It is an open-source software library [Chen, Guestrin, 2016]. This algorithm became widely used recently because it has been dominating applied machine learning tournaments and Kaggle competitions. The key advantages of this method are its speed and accuracy compared to similar methods. It is highly flexible and versatile for most machine learning tasks, such as classification, regression or ranking problems. For our model we used 300 estimators and we set the learning rate to 0.05.

LSTM (Long short-term memory) is a deep-learning technique. This model is a Recurrent Neural Network with a special architecture that allows it to avoid the problem of vanishing gradient and to learn long-term dependencies. The latter is especially useful for the task of text learning. We chose LSTM for our model because it is less dependent on a big training corpus compared to other popular neural network architectures. However, in order for LSTM to achieve good results a big training corpus is still needed. In our work we encountered this problem by trying to train the LSTM model on texts simply vectorized through the tf-idf technique. Because our corpus is relatively small, the training was not successful. In order to compensate for this, we implemented semantic vectors from the fastText database [Bojanowski et al. 2016]. These vectors also include information about the inner structure of the word. This was done by training the model using character n-gram features. For our LSTM network we used the Adam optimizer. Our model trained for 500 epochs with batch size 14.

4. Evaluation

Although overall accuracy is the easiest and most intuitive metric for model evaluation, it is quite useless for the purposes of evaluating multiclass classification. So instead we used precision, recall and F_1-score metrics. These metrics allowed us to understand how well our model can detect different genres and how good it is at distinguishing genres between each other. We also used the F_1-score metric that summarizes the results of precision and recall evaluation.

We tested our model with all of the methods and features described in the previous section. For overall results of the performance of all models see Table 3.

Table 3: Different classifiers' overall performance

Classifier + Feature	Precision	Recall	F_1-score
SVM + Words (tf-idf)	0.91	0.93	0.91
Logistic regression + Words (tf-idf)	0.90	0.93	0.91
XGBoost + Words (tf-idf)	0.90	0.93	0.91
LSTM + fastText vectors	0.87	0.88	0.87
SVM + Character trigram (tf-idf)	0.92	0.94	0.92
Logistic regression + Character trigram (tf-idf)	0.90	0.93	0.91
XGBoost + Character trigram (tf-idf)	0.91	0.94	0.91

As the result of the evaluation of our model, we see that the three best performing models are the SVM classifier trained on character trigrams, the XGBoost classifier also trained on character trigrams and the SVM trained on words. We also tested these three models for the performance of classification for each value that can be assigned during the classification in terms of FTD. In Tables 4–6 we compare the collective result for each FTD value across all 18 genres.

Table 4: The performance of the SVM + character trigram model for each FTD value

FTD value	Precision	Recall	F_1-score
0	0.94	0.99	0.97
1	0.22	0.07	0.11
2	0.72	0.27	0.39

Table 5: The performance of the XGBoost + character trigram model for each FTD value

FTD value	Precision	Recall	F_1-score
0	0.94	1.00	0.97
1	0.00	0.00	0.00
2	0.76	0.18	0.29

Table 6: The performance of the SVM + words model for each FTD value

FTD value	Precision	Recall	F_1-score
0	0.94	0.99	0.96
1	0.29	0.07	0.11
2	0.57	0.16	0.25

We also tested how well our best performing model can classify each genre. For each of the 18 Functional Text Dimensions we computed overall precision, recall and F_1-score. Each overall score is calculated accordingly to the share of each FTD value.

Table 7: The performance of the SVM + character trigram model for each FTD

FTDs	A1	A3	A4	A5	A6	A7	A8	A9	A11
Precision	0.79	0.96	0.96	0.98	1.00	0.84	0.84	0.96	0.87
Recall	0.79	0.95	0.96	0.97	0.98	0.85	0.87	0.96	0.93
F_1-score	0.72	0.95	0.95	0.97	0.99	0.81	0.85	0.95	0.90
FTDs	**A12**	**A13**	**A14**	**A15**	**A16**	**A17**	**A18**	**A19**	**A20**
Precision	0.90	0.99	0.88	0.98	0.72	0.85	1.00	1.00	0.99
Recall	0.95	1.00	0.94	0.97	0.77	0.92	0.98	0.98	0.98
F_1-score	0.93	0.99	0.91	0.97	0.72	0.88	0.99	0.99	0.99

Our model shows great results for some FTDs. However, this can be due to the low representability in the test corpus for these FTDs. Thus, the results for FTDs that have total value less than 10 in Table 2 can be interpreted as a majority class classification.

The best overall performing model is the SVM + character trigram. It shows the best result for the overall precision evaluation and it shares the first place for the overall recall. It can identify different FTD values and it also shows good performance for the one of the most represented A8 dimension (hardnews) (see Table 8). The zeros for the "1" FTD value are explained by the low number of "1" FTD values in our testing corpus.

Table 8: The performance of the SVM + character trigram model for the A8 dimension

FTD value	Precision	Recall	F_1-score
0	0.89	0.96	0.86
1	0.00	0.00	0.00
2	0.73	0.61	0.67

Classifiers that were tested in our research are "black-box" classifiers, which means that we do not know how the parameters for classification were set. However, the SVM classifier allows us to look at the most valuable features for each class, which can tell us a lot about the classification process. We analyzed our training corpus in terms of the most valuable features for determining each FTD. We did it using a SVM model that used character trigram as features. For the A4 (fiction) FTD the most valuable features were “قد”, “قال”, “ ها”. The first two are past tense markers and can also be interpreted as the elements of a narrative. “ ها” in its primary meaning is a feminine possessive pronoun, however it is also used in some complex structures, for example, relative clauses. For A7 (instruct) FTD the most valuable features are “إذا” and “إذ”. These are conjunctions that form conditional sentence. “لي”, “ي”, “نا” and “أنا ” were the features with highest weights in the A11 (personal) FTD. All of these features represent different variations of the pronoun “I”. The markers of A8 (hardnews) FTD are verb “قال” (to say) and conjunction “ان” that usually follows that verb to form indirect speech. The top features of the A9 (legal) FTD were “حاد”, “تحا” and “اتح”, which form one word “اتحاد” (union).

5. Analysis of results

5.1. Quality of detection

There are several conclusions that emerge from the results of our experiment. First, all of the models are pretty good at detecting when some text is not represented in a FTD, and thus has the value of “0”. All of them also have over 90% in precision and recall metrics for this FTD value. The key factor to the good performance of the SVM and XGBoost models was that they managed to correctly detect the FTDs with the value of “2” for the majority of texts. These models achieved high results for the precision metric, but they did not perform as well for the recall metric. Thus, our models perform well for the task of distinguishing texts that belong to different FTDs, but they are not as good at detecting all texts that belong to one FTD.

The sad conclusion that comes from the results of our experiment is that no models were able to correctly identify the “1” FTD values. The best algorithm for sensing the middle values of a genre was the algorithm that used LSTM with fastText word vectors. This model also showed good results in the recall metric for the FTD value of “2”, so it could adequately identify when a text pertained strongly to a FTD. However, the overall performance of the LSTM model was rather disappointing. It did not show good results for distinguishing texts between genres, which was the reason that this model got low scores.

The best performing feature was the character trigram; this corresponds to the results in [Sharoff et al, 2010]. However, it is hard to compare the overall results of our classification with other works, because to our knowledge no such research was conducted on Standard Arabic material. The results of the classification also depend greatly on the collection of texts used in the experiment.

5.2. Common problems

The main problem of our experiment is that our models did not achieve high scores for the recall evaluation for the non-zero values. A possible reason for this may be that for some texts the best performing classifiers did not assign a “2” value for the

FTD at all, which resulted in some texts being poorly scored. It is possible that the cause of this problem is that we used translated texts for the training and then tested our models on real texts from the Web.

Another problem that we encountered was the fact that we did not have a big training corpus. A bigger training corpus would likely have increased the results of most of the classifiers that we tested. A bigger corpus is definitely required for deep-learning techniques, such as LSTM. During our research we tried to train our LSTM model by simply vectorizing words in our corpus with the tf-idf technique, but because our corpus was relatively small, we were not able to do it.

6. Conclusions

In this paper we presented an experiment in which we built a model that classifies Arabic Web texts in terms of Functional Text Dimensions. For this experiment we produced a training corpus using machine translation tools. The original corpus and its annotations were taken from [Sharoff, 2018]. We also conducted an experiment to find out what features and what classifier has the best performance in this environment. The SVM classifier with character trigrams showed the best results. For testing the models, we collected a small representative corpus from Arabic websites with texts in Standard Arabic. They were annotated in terms of Functional Text Dimensions. The best performing model achieved precision of 93% of correctly identified FTD values. This indicates that the genre features are well preserved through Machine Translation.

Further work is still necessary, as we encountered several problems throughout our research. One of the possible directions of further work concerns building a bigger training corpus with more diverse texts in terms of their functional categories. A larger corpus would allow us to experiment with deep-learning techniques, such as the different architectures of the Recurrent Neural Network and the Convolutional Neural Networks. Another interesting development for our research could be the comparison of our classification to the classification of other corpora of Standard Arabic.

Experiments with semi-supervised settings for LSTM show promising results for the task of text classification [Johnson and Zhang, 2016]. We need to implement this technic in our future work.

We also need to increase the representativeness of our testing corpus. So far, a substantial part of it can be classified using only 3–4 dimensions. We need to collect and annotate more texts from personal blog-entries, opinion columns, product reviews, etc.

In our research we have seen that our model is capable of achieving reasonably good results. However, it still is not reliable with respect to recall by failing to detect many texts that belong to the same Functional dimension. This means that more testing and better machine learning techniques are needed.

References

1. *Adamzik K.* (1995), Textsorten—Texttypologie, Eine kommentierte Bibliographie, Nodus, Münster.
2. *Baroni M., Bernardini S., Ferraresi A., and Zanchetta E.* (2009). The WaCky wide web: a collection of very large linguistically processed web-crawled corpora, Language Resources and Evaluation, 43(3), pp. 209–226.
3. *Bojanowski P., Grave E., Joulin A., Mikolov T.* (2016), Enriching Word Vectors with Subword Information.
4. *Chen T., Guestrin C.* (2016), XGBoost: A Scalable Tree Boosting System, In 22nd SIGKDD Conference on Knowledge Discovery and Data Minin.
5. *Forsyth R., Sharoff S.* (2014), Document dissimilarity within and across languages: a benchmarking study, Literary and Linguistic Computing, 29, pp. 6–22.
6. *Görlach M.* (2004), Text types and the history of English, Walter de Gruyter.
7. *Hou H., Han P., Cao D.* (2015), The Application Based on Decision Tree SVM for Multi-class Classification.
8. *Johnson R., Zhang T.* (2016), Supervised and semi-supervised text categorization using LSTM for region embeddings, in ICML.
9. *Pedregosa et al.* (2011), Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825–2830.
10. *Piperski, A., Belikov, V., Kopylov, N., Selegey, V., and Sharoff, S.* (2013), Big and diverse is beautiful: A large corpus of Russian to study linguistic variation, In Proc 8th Web as Corpus Workshop (WAC-8).
11. *Santini, M., Mehler, A., and Sharoff, S.* (2010), Riding the rough waves of genre on the web, Genres on the Web: Computational Models and Empirical Studies, Springer, Berlin/New York.
12. *Sharoff S.* (2018), Functional text dimensions for annotation of web corpora, Corpora.
13. *Sharoff, S., Wu, Z., and Markert, K.* (2010). The Web library of Babel: evaluating genre collections. In Proc. of the Seventh Language Resources and Evaluation Conference.
14. *Trevilla L.* (2009), Ethnologue: Languages of the World.
15. *Zhang X., Zhao J., LeCun Y.* (2015), Character-level convolutional networks for text classification, In Advances in Neural Information Processing Systems, pp. 649–657.