Two Attempts to Predict Author Gender in Cross-Genre Settings in Dutch

Eduardo Brito, Rafet Sifa, and Christian Bauckhage

Fraunhofer IAIS, Fraunhofer Center for Machine Learning, Sankt Augustin, Germany

Abstract. This paper describes the systems designed by the Fraunhofer IAIS team at the CLIN29 shared task on cross-genre gender detection in Dutch. We show two alternative classification approaches: a rather standard one consisting of feature engineering and a random forest classifier; and an alternative one involving a LSTM classifier. Both are enhanced by a LDA model trained on stems. We considered various features such as frequency of function words, parts-of-speech and sentiment among others. We achieved 53.77% average accuracy in the cross-genre settings.

1 Introduction

The CLIN29 shared task is defined as predicting the gender of the author of collections of texts in Dutch. The organizers provided a labeled dataset from three different genres according to their source: "Twitter", "Youtube" and "News". For each of these genres, we were asked to predict the gender of the respective author by means of two different models: one trained with the text collection of the same genre (in-genre setting) and one trained with anything but text from the same genre (cross-genre setting).

We decided to design a common machine learning pipeline (per presented system) that works for all the settings. Thus, all the models are learned by applying the same pipeline and they only differ by their respective training data. This pipeline is described in section 3. Due to the shared task motivation of exploring approaches for cross-genre settings, we focused on tuning the pipeline components by evaluating each configuration on the three cross-genre settings exclusively, virtually ignoring intra-genre evaluations for hyperparameter tuning.

Finally, our experiments suggested that two different architectures were the best to tackle this shared task: one based on term frequency of function words and topic modelling; and another based on recurrent neural network using features of different nature. Since we were allowed to submit two different runs per setting, we provided predictions from both systems for each of the defined

settings. Although both approaches reached similar accuracy values in the cross-validations that we performed on the training dataset, the evaluation on the final test dataset shows that our first approach can generalize better on the evaluated genres.

2 Related Work

There is already a certain tradition on research on the broader area of author profiling, which includes detection of other author characteristics of the author such as their age, as we can see at the PAN shared task series[1]. However, most of the research has been performed for text in English. In particular, we could only find previous work on cross-genre author gender detection in Dutch at the PAN 2016 shared task (Rangel et al., 2016). Most of the features that our models use are inspired by the best performing systems presented for that challenge.

Syntactic features such as part-of-speech (POS) and syntactic dependency relations can predict author gender (Company, 2016). Due to their independence from the semantic content, we considered them suitable for cross-genre settings. Hence, we incorporated both of them in the second approach that we present.

Chen et al. (2018) showed that emoji usage in twitter highly depends on the gender of the author. However, emojis appear mostly on social media and they hardly exist in the "News" genre. Hence, we decided to remove all emojis and emoticons in order to prevent overfitting in cross-genre settings: our aim was to test the same set of features in all settings at the cost of eventually underperforming at in-genre settings.

3 Experimental Setup

Our two systems consist of three main processes: topic modeling, feature extraction and classification. An overview of the pipeline can be visualized in figure 1.

3.1 Topic modeling

Data: The topic modeling module takes two external corpora as input:

- **NLCOW14**[2]: the Dutch web corpus from the COW initiative (Schfer and Bildhauer, 2012; Schfer, 2015).
- **Wikipedia**: A Dutch Wikipedia dump from 2018-10-01[3].

Filtering: In order to comply with the cross-genre setting rules of not using any text coming from the genre where it is evaluated, we filtered out all documents from NLCOW14 whose URL contains the string "twitter" or "youtube".

[1] https://pan.webis.de
[2] `https://github.com/rsling/cow` , `https://github.com/rsling/texrex`
[3] `https://linguatools.org/tools/corpora/wikipedia-monolingual-corpora/`

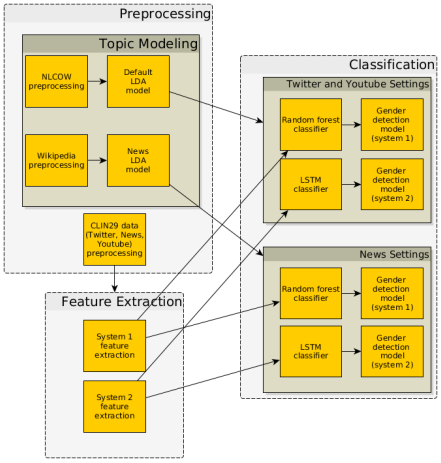

Fig. 1. Pipeline overview including both presented systems.

Since the concept of *genre* is rather defined by the data source than by the text content, we consider that this approach suffices to discard all texts belonging to the "twitter" and "youtube" genres: each of these genres comes from a single data source.

In contrast to the other genres, "News" may come from diverse data sources which we cannot reliably filter out by just checking the document URL against a blacklist in the same way that we did for the NLCOW14 corpus. Therefore, we selected the Wikipedia corpus for the topic modelling part of the two "News" settings.

Preprocessing: From each document, a set of words is filtered out, namely: stop words, emoticons, emojis, URLs and words starting with the characters '#'and '@' (typical for Twitter hashtags and mentions respectively). Then, each word token is both lemmatized and stemmed consecutively. Word tokenization and lemmatization are performed with the help of the Frog parser (Bosch et al., 2007) available via LaMachine[4] while stemming is achieved with NLTK (Bird et al., 2009).

Model training: We train latent Dirichlet allocation (LDA) models (Blei et al., 2003) with 50 topics on a document basis using the respective Gensim software package (Řehůřek and Sojka, 2010). That is, each document is assigned a 50-dimensional vector at the end of this process.

3.2 Feature extraction

The labeled data from the three genres goes through a similar preprocessing as described for topic modeling. However, instead of being lemmatized and

[4] `https://proycon.github.io/LaMachine/`

stemmed, non-function words are normalized into some special tokens. Namely, words which are parsed as nouns, verbs, adjectives and special words are substituted by a token related to their POS. Additionally, tokens recognized as ordinals, hours, dates and other types of numbers are normalized to respective tokens as well.

The feature vector used for classification is dependent on the system:

System 1 We create feature vectors based on function words. In our preprocessed text, only function words and normalized tokens are left. For each document, we compute the logarithm of the term frequency vector normalized with respect to the $L1$-norm as proposed by Diederich et al. (2003).

Since we assume that their usage frequency vary depending on the context where they appear, we concatenate a 50-dimensional vector inferred from the trained LDA model in order to provide contextual information to the model.

System 2 We decided to try also a neural network architecture based on a long short-term memory (LSTM) classifier (Hochreiter and Schmidhuber, 1997). Since some dependencies across words within a document may also be characteristic from a specific gender (Chen et al., 2018), the LSTM should capture their relevancy for classification. For this architecture, we understand a document as a sequence of words. We transform each document into a sequence of vectors relative to the word tokens of each document. Per sentence, we also add two additional tokens marking the start and the end of the sentence respectively.

Each of these vectors contains:

- A coarse-grained POS tag (e.g. verb).
- A fine-grained POS tag (e.g. 1st person, singular, present).
- Position of the word within the sentence.
- Relative position of the word (position / sentence length)
- Position distance of the word respect to the parent word (in the syntax dependency tree) within the sentence.
- Polarity score.
- Subjectivity score.

For each vector, we append a 50-dimensional vector inferred by the respective LDA model from the document where the respective word resides in the same fashion that we do for system 1.

The POS tags are based on the *Corpus Gesproken Nederlands* (CGN) tags (Van Eynde, 2004) obtained by the Frog parser. For each parsed CGN tag, we split the string into the part before the brackets, which is our coarse-grained POS (e.g. "WW" for verbs); and the part within the brackets, which constitutes our fine-grained POS tag (e.g. "pv,tgw,ev" for finite verb, present tense, singular).

The polarity and subjectivity scores are obtained from the Pattern software package (Smedt and Daelemans, 2012).

In order to have a document representation of fixed size, we restrict each representation to a sequence of 100 vectors. If a document has less than 100 vectors, we prepend zero vectors until we reach the fixed size.

Setting	Max depth
Twitter in-genre	5
Twitter cross-genre	4
News in-genre	11
News cross-genre	8
Youtube in-genre	10
Youtube cross-genre	5

Table 1. Max depth length set for the random forest classifiers for each setting.

3.3 Classification

For both systems, the training data depends on the evaluated setting: for the in-genre settings, only the provided examples from the same genre are used to generate the feature vectors; for the cross-genre settings, the examples for the not evaluated genres are fetched. Additionally, each feature vector includes a topic vector inferred by the respective topic model. For instance, the training set the for "In-genre Twitter" setting consists of the preprocessed "Twitter" examples plus a topic vector derived from a topic model trained on the NLWAC corpus, whereas the training set for the "Cross-genre Twitter" setting is made out of the preprocessed "News" and "Youtube" examples (plus the respective topic vectors).

System 1 We train a random forest classifier (Breiman, 2001) for each of the defined settings by means of its scikit-learn implementation (Pedregosa et al., 2011) on the generated vectors from the feature extraction module and the labels from the given dataset. All classifiers have identical hyperparameter configurations (namely 2001 estimators, log2 of the number of features as maximum features, minimum number of samples required to split an internal node, 2 minimum number of samples required to be at a leaf node) but a different maximal depth of the trees, which is detailed in table 1. We found no significant improvement by having other setting-specific hyperparameters.

System 2 Per setting, we train the following recurrent neural network architecture with Keras[5]:

- A bidirectional LSTM layer with 50 hidden units with 0.5 dropout ratio for the linear transformations of the inputs and the recurrent state respectively.
- A dropout layer with 0.5 ratio
- A softmax layer

The architecture is trained by a Nesterov Adam optimizer (Dozat, 2016) with a categorical crossentropy loss function.

[5] `https://keras.io/`

Setting	# Training examples	Sys. 1	Sys. 2
In-genre			
Twitter	20,000	59.45	59.15
News	1,832	50.30	49.40
Youtube	14,744	55.66	55.11
Average		**55.14**	54.55
Cross-genre			
Twitter	16,576	54.25	51.77
News	34,744	54.80	50.40
Youtube	21,832	52.27	50.94
Average		**53.77**	51.04

Table 2. Accuracy (in %) of our submitted predictions from the two presented systems.

4 Results and discussion

As we can see from table 2, system 1 performs consistently better than system 2 in all settings. This is an unexpected result considering that the latter learns from a wider range of features. It is possible that the amount of training examples might have not sufficed for the second approach to work better than the first one because LSTM classifiers require considerably more labeled examples than random forest classifiers.

Another surprising result is that the accuracy in the cross-genre setting is higher than in the in-genre one in the case of the "News" datasets. This is probably to the much larger size of the training dataset for the cross-genre setting (34,744 examples from "Twitter" and "Youtube") than for the in-genre setting (1,832 examples from "News"). Therefore, system 1 may still perform better if it had a larger training dataset.

Our models show that they can only predict author gender for only a (small) part of the evaluated text collections. Hence, we suspect that most pieces of text do not have any genre-independent characteristics that can be reliably used to predict the gender of the author, although in some of them this may be possible. Therefore, we remain uncertain about whether it is feasible (in general) to predict author gender by using features that are independent from the text genre.

Bibliography

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3(Jan):993–1022.

Antal van den Bosch, Bertjan Busser, Sander Canisius, and Walter Daelemans. 2007. An efficient memory-based morphosyntactic tagger and parser for Dutch. *LOT Occasional Series* 7:191–206.

Leo Breiman. 2001. Random forests. *Machine learning* 45(1):5–32.

Zhenpeng Chen, Xuan Lu, Wei Ai, Huoran Li, Qiaozhu Mei, and Xuanzhe Liu. 2018. Through a gender lens: Learning usage patterns of emojis from large-scale android users. In *Proceedings of the 2018 World Wide Web Conference*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, WWW '18, pages 763–772.

Juan Soler Company. 2016. Use of discourse and syntactic features for gender identification. In *STAIRS*. pages 215–220.

Joachim Diederich, Jörg Kindermann, Edda Leopold, and Gerhard Paass. 2003. Authorship attribution with support vector machines. *Applied intelligence* 19(1-2):109–123.

Timothy Dozat. 2016. Incorporating nesterov momentum into adam .

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Francisco Rangel, Paolo Rosso, Ben Verhoeven, Walter Daelemans, Martin Potthast, and Benno Stein. 2016. Overview of the 4th author profiling task at PAN 2016: cross-genre evaluations. In *Working Notes Papers of the CLEF 2016 Evaluation Labs. CEUR Workshop Proceedings/Balog, Krisztian [edit.]; et al.*. pages 750–784.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, pages 45–50.

Roland Schfer. 2015. Processing and querying large web corpora with the COW14 architecture. In Piotr Baski, Hanno Biber, Evelyn Breiteneder, Marc Kupietz, Harald Lngen, and Andreas Witt, editors, *Proceedings of Challenges in the Management of Large Corpora 3 (CMLC-3)*. UCREL, IDS, Lancaster.

Roland Schfer and Felix Bildhauer. 2012. Building large corpora from the web using a new efficient tool chain. In Nicoletta Calzolari (Conference

Chair), Khalid Choukri, Thierry Declerck, Mehmet Uur Doan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. European Language Resources Association (ELRA), Istanbul, Turkey, pages 486–493.

Tom De Smedt and Walter Daelemans. 2012. Pattern for python. *Journal of Machine Learning Research* 13(Jun):2063–2067.

Frank Van Eynde. 2004. Part of speech tagging and lemmatizing of the Corpus Gesproken Nederlands (Spoken Dutch Corpus). *KU Leuven* .