Reusable workflows for gender prediction

Matej Martinc, Senja Pollak
Jožef Stefan Institute
Jamova cesta 39, Ljubljana, Slovenia
{matej.martinc, senja.pollak}@ijs.si

Abstract

This paper presents a system for author profiling (AP) modeling that reduces the complexity and time of building a sophisticated model for a number of different AP tasks. The system is implemented in a cloud-based visual programming platform ClowdFlows and is publicly available to a wider audience. In the platform, we also implemented our already existing state of the art gender prediction model and tested it on a number of cross-genre tasks. The results show that the implemented model, which was trained on tweets, achieves results comparable to state of the art models for cross-genre gender prediction. There is however a noticeable decrease in accuracy when the genre of a test set is different from the genre of the train set.

Keywords: author profiling, workflows, reusability and replicability

1. Introduction

In recent years, the data from the user-generated content (UGC) has become a popular resource for studies in natural language processing (NLP) and text mining, especially due to its accessibility, size and a near real-time publishing. Messages from the social media by personal users display a trend towards more personal content (Naaman et al., 2010), which makes it a useful dataset for learning about the users' characteristics, related to demographics, psychological profile or mental health. The field that uses text documents to discover users' attributes automatically is known as author profiling (AP) and includes tasks, such as the prediction of author's gender (Rangel et al., 2017), which is also at the core of this paper, but also the prediction of author's age (Rangel et al., 2016), personality type (Verhoeven et al., 2016) or language variety (Rangel et al., 2017).

While deep learning approaches are gradually taking over different areas of NLP, the best approaches to AP still use more traditional classifiers and require extensive feature engineering (Rangel et al., 2017). This test-driven approach relies heavily on trying out different feature and parameter configurations, which can be very time consuming. To accelerate this process of model building, we propose the implementation of a system that enables faster model prototyping by employing the visual programming (VP) paradigm. The VP paradigm simplifies the representation of complex procedures into visual arrangements of their building blocks and consequently enables faster and more efficient model prototyping and model fine-tuning.

The contribution of this paper is two-fold and is anchored in recent trends in reusability and replicability in NLP. Our AP system implementation is built in an online visual programming environment, which is to the best of our knowledge the first of its kind. The proposed approach to constructing prototype AP models through visual workflows of NLP components contributes to the reusability in NLP on the level of specific components, as well as on the level of the entire gender prediction models. The pretrained models for five languages (four of which we developed for PAN 2017 (Martinc et al., 2017)) can be reused in new experiments and tested on new datasets. This has been showcased in cross-genre experiments in Section 4, which also show that our model, trained on tweets, is robust and achieves results comparable to the models built specifically for the cross-genre task. On the other hand, specific components and the entire workflow can be reused for building new models for AP as well as for other text classification tasks. Even if the PAN community already encourages code sharing (`https://github.com/pan-webis-de`), the simplified visual representation reduces the technical skills required for building complete systems and opens the experimentation also to the users outside of the programming community (e.g., linguists) which is very important in such an interdisciplinary field as AP. The experimentation is also easier/faster, since no specific software installation is required.

By publishing the workflows of our experiments, this work contributes to the paradigm of transparent experimentation and replicability. Since our experiments are proposed as online workflows, they can be easily reproduced and used for comparison of results.

2. Background technologies and related work

The platform used for the implementation of the system for rapid prototyping, implementation and testing of models is ClowdFlows (Kranjc et al., 2012), (`http://clowdflows.org`). ClowdFlows is a cloud-based web application for composition, execution and sharing of interactive data mining workflows.

ClowdFlows enables visual programming and has a GUI in which a user connects processing components (i.e. widgets) into executable pipelines (i.e. workflows) on a design canvas. It has a web based GUI for building workflows, runs in all major browsers and requires no installation.

ClowdFlows visual programming platform was chosen for the implementation of the system for AP modeling because of its open source nature that allows anybody to expand its widget stack and because of its system for publishing workflows together with the data, allowing reproducibility of our results. None of the considered open source alternatives, such as the GATE platform (Cunningham, 2002)

(General Architecture for Language Engineering) or the open source implementation of the IBM's Unstructured Information Management Architecture (Apache UIMA) (Ferrucci and Lally, 2004) enable workflow and data sharing. TextFlows (Perovšek et al., 2016), a fork of ClowdFlows specialized for NLP and text mining workflows, was also initially considered for the implementation of the AP modeling system. The reason why this platform was not chosen were its format restrictions, or more specifically, the enforced Annotated Document Corpus (ADC) format, which would increase the memory consumption of the system, prolong the execution time and put heavy restrictions on the size of the input corpora.
Even if—to the best of our knowledge—the AP tools have not yet been made available in form of executable workflows, the field of AP has a strong research tradition. Approaches for predicting author's age (Rangel et al., 2016), language variety (Rangel et al., 2017) or personality type (Verhoeven et al., 2016) have been proposed, while gender prediction has the strongest tradition. The earliest attempts at AP that covered gender identification started with Koppel et al. (2002), who used parts of the BNC, but continued on other corpora, such as the ACL corpus of scientific papers (Vogel and Jurafsky, 2012) or UGC corpora, esp. in the context of PAN shared tasks (Rangel et al., 2017). This paper also builds upon our contribution to the PAN 2017 shared task (Martinc et al., 2017), but decomposes the approach to the reusable building blocks, makes the models available for new experimentation and tests the model on cross-genre settings.

3. Widgets for rapid model prototyping

We implemented new tools in the ClowdFlows platform that would enable its users a fast test driven development in the field of AP. These tools are presented as graphical components (widgets) that can be connected into workflows and range in functionality from tools for corpus representation and NLP to more general classification tools. This section briefly describes the newly implemented tools and how they can be combined into a workflow for building state of the art classification models:

Load Corpus From CSV widget takes a comma-separated values (CSV) file containing the corpus with texts and annotations and converts it into a Pandas dataframe (McKinney, 2011) that enables further processing. Lines of the dataframe present documents and columns present different text annotations or the texts itself. The same format applies to the input CSV file. The first line of the loaded CSV file should also contain names of the columns.

Select Corpus Attribute widget takes a corpus in a dataframe form as an input and returns only the column of the dataframe defined by the user. This enables extracting only the part of the corpus that is needed, reducing the computational complexity of further processing. The column is returned as a Python list.

Remove Punctuation widget takes a column of the corpus (in the form of a Python list), usually the one containing texts, and returns a list of texts without the most common punctuation characters.

Remove Stopwords widget takes a column of the corpus containing texts and returns a list of texts without stopwords. The user can choose between stopwords lists for English, Portuguese, Slovenian or Spanish.

Tweet Cleaner widget takes a column of the corpus containing texts and by default replaces hashtags, mentions and URL-s with filler tokens (HTTPURL, TWEETMENTION and HASHTAG). Hashtags, mentions and URL-s can also be removed if the user chooses the *remove mode* in the widget parameters.

Affix Extractor widget takes a column of the corpus as an input and returns token affixes. The user can define the length and type of affixes (prefixes, suffixes and punctuation affixes, i.e. *beg-puncts* (Sapkota et al., 2015).

Count Patterns widget takes a column of the corpus containing texts and counts the specified patterns in the text. The user can specify the patterns as a comma separated list of word tokens or phrases, or select from already defined patterns, such as character floods and emojis. The user can also select if he wants raw counts or counts that are normalized by the number of characters in the text. By default, the widget returns a list containing the counts for each text in the corpus, however the widget can also return an integer representing the count of a specific pattern for the whole corpus.

Emoji Sentiment widget (Novak et al., 2015) takes a column of the corpus containing texts as an input and returns a sentiment score for every text. The sentiment is calculated as the sum of sentiment values of all the emojis in the text (no normalization is used).

TF-IDF Vectorizer widget takes a text column of the corpus as an input and returns a vector of calculated TF-IDF weights. The user can choose the type of tokens for which he wants the TF-IDF values, such as word, character and word bound character n-grams with different n values. Minimum and maximum document frequency of the token can be defined, meaning that all the tokens not fitting in the desired interval will be ignored. The tokens can optionally be lowercased before the TF-IDF calculation and different versions of TF-IDFs can be calculated (with smooth IDF and sublinear TF). The widget relies heavily on the TF-IDF vectorizer implemented in the Scikit library (Pedregosa et al., 2011).

Feature union widget takes heterogeneous features and a target value as inputs and returns a dataset containing the union of the features that can be feed as an input to different Scikit classifiers. The user can define weights for different types of features which influence the penalization of these features during the classifier learning phase.

Cross Validation widget takes a classifier and a dataset produced by a *Feature union* widget as an input

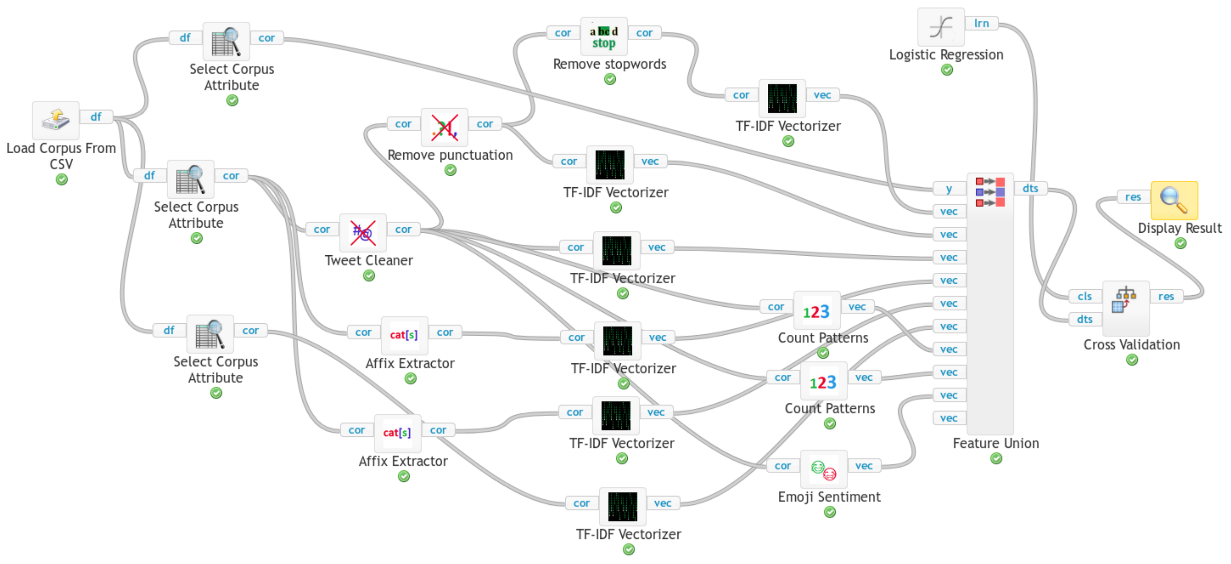


Figure 1: The model for gender classification implemented in the ClowdFlows platform (publicly available at `http://clowdflows.org/workflow/10620/`).

and conducts cross-validation of the classifier on the dataset. The user can define the number of folds and the metrics for evaluation (accuracy, precision, recall and f-score).

Display result widget is a simple widget that displays the results of the cross-validation test.

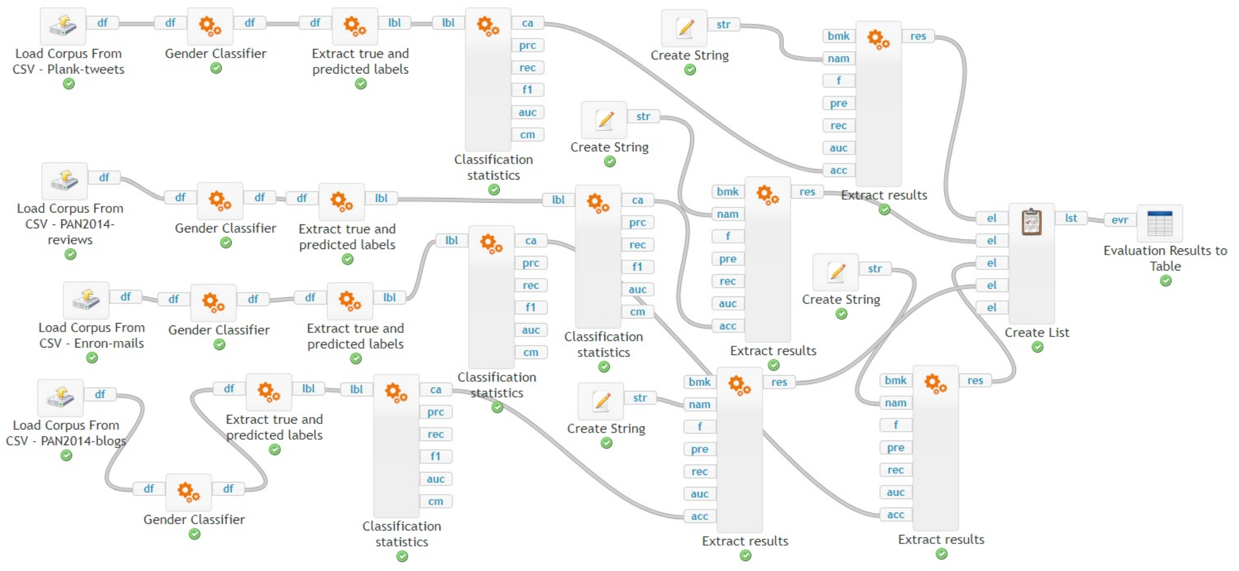
Besides the above mentioned widgets, ClowdFlows also contains different Scikit classifiers (Pedregosa et al., 2011) that can be used in model creation, such as Naive Bayes classifier, Logistic regression, Support Vector Classification, Linear Support Vector Classification, K-nearest neighbour classifier and Decision tree classifier. All these classifiers have specific parameters that can be tweaked for better performance.
The workflow in Figure 1 presents how we can combine the developed NLP widgets into a workflow that trains state of the art gender classification model. The workflow structure is nearly identical to the structure of our model used in the PAN competition (Martinc et al., 2017). We load the subcorpus[1] of the PAN 2017 corpus containing documents by 500 authors (each document contains around 100 concatenated tweets written by the author), part of speech (POS) tags and labeled gender from the CSV file by the *Load Corpus From CSV* widget. We use three *Select Corpus Attribute* widgets to extract text, gender labels and POS tags. Gender labels are directly connected to the *Feature Union* widget as a target value, while we do some further processing on text and POS tags. POS tags are TF-IDF weighted and converted into POS trigrams with the help of the *TF-IDF Vectorizer*. For text we build three levels of preprocessing (in the first level we just remove mentions, hashtags and URL-s, in the second level we additionally remove punctuation and in the third level we also remove stopwords), which are all later TF-IDF weighted. We also build different affix features with the *Affix extractor*, pattern counts features (character flood and emoji counts) created by *Count pattern* widgets and calculate document sentiment. All these heterogeneous features are combined in the *Feature Union* widget which builds a dataset. This dataset is used in the cross-validation experiments using Logistic Regression as a classifier and accuracy as a measure. The results are displayed in the *Display Result* widget.
Since this workflow is publicly available (`http://clowdflows.org/workflow/10620/`) it can be used as a template for building and testing models for many different AP tasks. Going to the published url of the workflow and clicking on the *Launch workflows! (Create a copy)* button generates a copy of the workflows together with its data, that a user can change without affecting the original workflow. This way, the published workflow can be modified to better fit some other AP task and data by, for example, adding/removing features and tweaking parameters, which is fast and simple because of the visual programming paradigm.

4. Model Reusability

Fast prototyping is important for the users who wish to develop new models quickly. On the other hand, there are users who just wish to use an already developed and trained model. For example, if we have a dataset and we want to annotate it with the gender of their author, pretrained models can be an optimal solution. For this reason we took our model for gender classification developed during the PAN 2017 shared task[2] and packed it in a widget *Gender Classifier* (for details see (Martinc et al., 2017)). The widget takes

[1]The number of tweets was limited to 50,000 to comply with the Twitter terms of use: `https://developer.twitter.com/en/developer-terms/agreement-and-policy`.

[2]The model achieved the second highest overall accuracy (for English gender prediction, the accuracy was 80,71%).



Load Corpus From CSV - Plank-tweets
Gender Classifier
Extract true and predicted labels
Classification statistics
Create String
Extract results
Load Corpus From CSV - PAN2014-reviews
Load Corpus From CSV - Enron-mails
Load Corpus From CSV - PAN2014-blogs
Create List
Evaluation Results to Table

Figure 2: The workflow for cross-genre experiments conducted in the ClowdFlows platform (publicly available at `http://clowdflows.org/workflow/11042/`).

a Pandas dataframe as input and returns a dataframe with an additional column with predicted gender labels. The user needs to define the name of the column containing text documents as a parameter and choose the language of the text (English, Spanish, Portuguese, Arabic or Slovenian).

Even if the gender classification model was developed for the classification of tweets, we tested it not only on a separate corpus of tweets, but also in a cross-genre classification. The PAN 2016 AP shared task (Rangel et al., 2016)—and additional research conducted by part of the winning team (Medvedeva et al., 2017)—dealt with cross-genre classification, so we are able to compare the results that our model achieved on corpora of blogs and reviews with the other state of the art models built specifically for cross-genre performance.

Figure 2 shows the workflow (available at `http://clowdflows.org/workflow/11042/`) for the cross-genre experiments conducted on four corpora from different genres of UGC with labeled gender.

PAN2014-blogs (Rangel et al., 2014)[3]: the dataset consists of 2,278 blogs by 146 authors (73 male and 73 female). The blogs of the same author were concatenated to a single document, representing one author.

PAN2014-reviews : the PAN 2014 AP train set contains hotel reviews from 4,160 authors (2,080 male and 2,080 female). As above, the reviews of the same author were grouped to a single document.[3]

Plank-tweets (Plank and Hovy, 2015)[4]: dataset consists of 561 male and 939 female authors and is already preprocessed in terms of retweet removal and concatenation of tweets by the same author. We removed 378 female authors to balance the two classes for easier comparison with other datasets that are also balanced.

Enron-mails dataset (Klimt and Yang, 2004)[5] contains 517,431 e-mails of Enron employees, 105 male and 43 female. In this corpus emails are arranged by folders for specific users. The body of each e-mail was parsed by removing the header and reply text. We balanced the corpus by removing 62 male authors and concatenated e-mails of the same author to a single document.

We loaded all four corpora from the CSV files and converted them into a Pandas dataframe. The gender was predicted by our *Gender Classifier* widget. We used *Extract true and predicted labels* widget for extracting true gender labels and the labels predicted by the classifier that are used to calculate different metrics – we were interested in the accuracy of the model for which we used the *Classification statistics* widget. Widget *Extract results* is used to properly format the output of the widgets *Classification statistics* and *Create String* (which is used for adding a name of the corpus for the final table representation). *Create list* groups all the results, which are then finally presented in the table form with the *Evaluation Results to Table* widget.

Results (Table 1) show that the genre of the train set matters. Our gender classification model, which was trained on tweets, proved far more accurate on tweets (Plank-tweets) than on other genres. Our model achieved very similar accuracies on blogs and reviews and achieved the lowest accuracy on emails. We can compare the results of our model to the results of the winners of the PAN 2016 AP shared task (Medvedeva et al., 2017). In the experiments where their model was trained on English tweets (train set from the PAN 2016 AP competition) they achieved 73.43% accuracy on PAN2014-blogs and 50% accuracy on PAN2014-reviews while mails were not one of their test corpora. The

[3]PAN2014 data: `http://pan.webis.de/data.html`

[4]`https://bitbucket.org/bplank/wassa2015/src`

[5]`https://www.cs.cmu.edu/~./enron/`

Corpus	Accuracy
PAN2014-blogs	65.75%
Plank-tweets	**74.69%**
Enron-mails	53.41%
PAN2014-reviews	63.73%

Table 1: Results of the cross-genre experiments (majority classifier for all the corpora is 50%). The model was trained on the tweets from the PAN 2017 AP training set and tested on four different genres (tweets, blogs, mails, reviews).

results show that our model—even if it was not built with cross-genre classification in mind—gives more constant accuracies for the two compared genres, achieving higher accuracy on reviews and lower on blogs.

5. Conclusion

This paper presents a system for rapid AP model prototyping together with the results of cross-genre gender classification experiments. The implemented system reduces the complexity and time of building a sophisticated model for a number of different AP tasks and also gives the user enough freedom and flexibility to allow for thorough exploration of different possibilities. We also demonstrate that our model for gender classification, that was developed for tweet classification in the PAN 2017 task, can be used on other genres and gives results that are comparable to the models built for cross-genre classification (we achieve higher results for reviews, but lower for blogs). There is however a noticeable decrease in accuracy when the genre of a test set is different from the genre of the training set. The presented work contributes to the replicability and reusability in NLP since all the components, models and experiments are available to the research community in the form of online workflows and widgets. In future work, the models for other languages will be tested on available datasets, while the constructed workflows will be used for other AP tasks.

Acknowledgements

The authors acknowledge the financial support from the Slovenian Research Agency through core research programme Knowledge Technologies (P2-0103) and the basic research project Resources, Tools and Methods for the Research of Nonstandard Internet Slovene (J6-6842). The authors also acknowledge the project ClowdFlows Data and Text Analytics Marketplace on the Web (CF-Web), which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 754549. We would also like to thank Iza Škrjanec and Katja Zupan for their contribution in the scope of the PAN 2017 shared task.

6. Bibliographical References

Cunningham, H. (2002). Gate, a general architecture for text engineering. *Computers and the Humanities*, 36(2):223–254.

Ferrucci, D. and Lally, A. (2004). Uima: an architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10(3-4):327–348.

Klimt, B. and Yang, Y. (2004). The Enron corpus: A new dataset for email classification research. In *European Conference on Machine Learning*, pages 217–226. Springer.

Koppel, M., Argamon, S., and Shimoni, A. R. (2002). Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, 17(4):401–412.

Kranjc, J., Podpečan, V., and Lavrač, N. (2012). ClowdFlows: A cloud based scientific workflow platform. In Peter A. Flach, et al., editors, *Proc. of ECML/PKDD (2)*, volume 7524 of *LNCS*, pages 816–819. Springer.

Martinc, M., Škrjanec, I., Zupan, K., and Pollak, S. (2017). Pan 2017: Author profiling-gender and language variety prediction. *CLEF 2017 Evaluation Labs and Workshop – Working Notes Papers*.

McKinney, W. (2011). Pandas: a foundational Python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, pages 1–9.

Medvedeva, M., Haagsma, H., and Nissim, M. (2017). An analysis of cross-genre and in-genre performance for author profiling in social media. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 211–223. Springer.

Naaman, M., Boase, J., and Lai, C.-H. (2010). Is it really about me?: Message content in social awareness streams. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative work*, pages 189–192. ACM.

Novak, P. K., Smailović, J., Sluban, B., and Mozetič, I. (2015). Sentiment of emojis. *PloS one*, 10(12):e0144296.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, 12:2825–2830.

Perovšek, M., Kranjc, J., Erjavec, T., Cestnik, B., and Lavrač, N. (2016). TextFlows: A visual programming platform for text mining and natural language processing. *Science of Computer Programming*, 121:128—-152.

Plank, B. and Hovy, D. (2015). Personality traits on twitter -or- how to get 1,500 personality tests in a week. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA)*. Lisbon, Portugal.

Rangel, F., Rosso, P., Chugur, I., Potthast, M., Trenkmann, M., Stein, B., Verhoeven, B., and Daelemans, W. (2014). Overview of the author profiling task at PAN 2014. In *CLEF 2014 Evaluation Labs and Workshop – Working Notes Papers*. CEUR.

Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., and Stein, B. (2016). Overview of the 4th Author Profiling Task at PAN 2016: Cross-genre evaluations. In *CLEF 2016 Working Notes*. CEUR-WS.org.

Rangel, F., Rosso, P., Potthast, M., and Stein, B. (2017). Overview of the 5th Author Profiling Task at PAN 2017:

Gender and Language Variety Identification in Twitter. In Linda Cappellato, et al., editors, *Working Notes Papers of the CLEF 2017 Evaluation Labs*, CEUR Workshop Proceedings. CLEF and CEUR-WS.org, September.

Sapkota, U., Bethard, S., Montes-y-Gómez, M., and Solorio, T. (2015). Not all character n-grams are created equal: A study in authorship attribution. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 93–102.

Verhoeven, B., Daelemans, W., and Plank, B. (2016). Twisty: A multilingual twitter stylometry corpus for gender and personality profiling. In *LREC*.

Vogel, A. and Jurafsky, D. (2012). He said, she said: Gender in the ACL anthology. In *Proceedings of the ACL-2012 Special Workshop on Rediscovering 50 Years of Discoveries*, pages 33–41. ACL.