# Classification of Historical Notary Acts with Noisy Labels

Julia Efremova[1], Alejandro Montes García[1], and Toon Calders[1,2]

[1] Eindhoven University of Technology, The Netherlands
[2] Université Libre de Bruxelles, Belgium

**Abstract.** This paper approaches the problem of automatic classification of real-world historical notary acts from the $14^{\text{th}}$ to the $20^{\text{th}}$ century. We deal with category ambiguity, noisy labels and imbalanced data. Our goal is to assign an appropriate category for each notary act from the archive collection. We investigate a variety of existing techniques and describe a framework for dealing with noisy labels which includes category resolution, evaluation of inter-annotator agreement and the application of a two level classification. The maximum accuracy we achieve is 88%, which is comparable to the agreement between human annotators.

## 1 Introduction

Text Classification (TC) is the problem of assigning one or several predefined categories to text documents [5]. TC is a relevant research question, given the large amount of uncategorized digital text documents. It is widely used to solve text mining problems (e. g. topic detection, spam filtering, folktale classification, news analysis, SMS mining, etc. [7,4,6]).

The TC has been studied by many researchers. Sebastiani [8] presented a detailed survey about supervised TC techniques. Later Ikonomakis [5] extended his work and summarised available machine learning approaches for the overall TC process. Recently Aggarwal [1] provided a survey of a wide variety of TC algorithms. Constantopoulos et al. [3] designed a digital library for historical documents that includes indexing techniques for the document annotation and retrieval.

In our case we have to deal with historical data and we use a number of machine learning algorithms together with the extraction of names, places and lexical information. Archived documents, presented in the form of unstructured text, contain a large amount of information about legal events. In many cases they are the only source of historical facts.

In this paper we develop of a classification framework for a large collection of Dutch notary acts from the $14^{\text{th}}$ to the $20^{\text{th}}$ century, examine the influence of lexical features, namely Parts-Of-Speech, as well as personal information elimination on the classification process, and provide an annotated corpus to the research community[1].

[1] `http://wwwis.win.tue.nl/~amontes/ecir2015/dataset.zip`

A. Hanbury et al. (Eds.): ECIR 2015, LNCS 9022, pp. 49–54, 2015.

## 2 Data Description and General Approach

Our dataset is comprised of notary acts provided by the Brabants Historical Information Center. The documents contain information about people involved in property transfers, loans, wills, etc. They were written between the $14^{\text{th}}$ and the $20^{\text{th}}$ century. An example can be found on `http://goo.gl/NhdFeq`. 115 967 documents out of 234 325 documents were labelled by volunteers with a single category for each document that describes their content. The assigned categories often contain spelling errors and duplicates and the collection is unbalanced.

The original dataset contains 455 categories identified by volunteers and around 20% of the classified documents belong to only one category.

To preprocess the documents we remove from the raw data punctuation marks or non-alphabetical symbols and transform the text to lower case. Then we split the original documents into sets of words called *tokens* and remove Dutch stopwords. We explore *personal information elimination* (PIE) by removing person names and locations. To do so we use person name and location dictionaries obtained from the database of the Meertens Institute[2] and the Historical Sample of the Netherlands[3]. Moreover, we apply stemming [1].

Then we create a feature for each remaining token, and set their values using the *term frequency inverse document frequency* (TF-IDF) [5]. The output of the feature extraction step is a set of numerical features. The entire vocabulary of the overall collection of notary acts is very large, the resulting feature set is sparse. Table 1 demonstrates the number of unique features for each experimental setup.

To overcome the sparsity problem we use different feature selection techniques, namely *Pearson's chi-squared test* [5] and *Latent semantic analysis* [1] and choose the 2000 most representative features for the whole corpus. In addition, we investigate the role of *part of speech (POS) lexical features*: nouns, verbs and adjectives. To obtain POS fragments we use the Frog tool[4] which is a morpho-syntactic tagger and parser for Dutch text [2].

The last step of the overall TC process is learning the model and classification. We apply and evaluate the *Support Vector Machines* (SVM) [1] classifier use from the scikit-learn python tool[5] with a linear basis kernel function. Then the algorithm is ready to classify the documents [1,5].

**Table 1.** The number of unique words-features in each experiment

| Stemming | Personal Information Elimination | Number of features |
|---|---|---|
| ✗ | ✗ | 49967 |
| ✗ | ✓ | 38670 |
| ✓ | ✗ | 42383 |
| ✓ | ✓ | 31106 |

[2] `http://www.meertens.knaw.nl/nvb/`
[3] `http://www.iisg.nl/hsn/data/`
[4] `http://ilk.uvt.nl/frog/`
[5] `http://scikit-learn.org/`

## 3 Dealing with Noisy Labels

To identify duplicated categories we generate pairs of categories which can be candidates for merging using a confusion matrix $\mathcal{M}$. Fig. 1 shows a part of the confusion matrix for eleven randomly selected categories. The complete $\mathcal{M}$ has 455 rows and columns. The confusion means that one category was incorrectly predicted as another category. The matrix is obtained by the SVM classifier applied to notary acts without stemming, PIE or feature selection (see Experiment 1, Section 5). We analyse the confusion matrix to identify categories that were duplicated and perform a category resolution with the help of an expert.

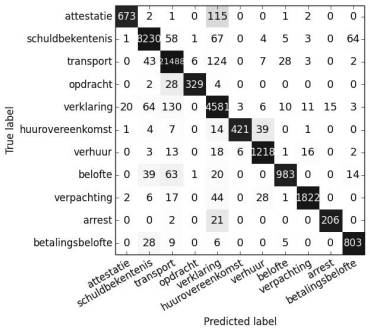



**Fig. 1.** Confusion matrix for randomly selected categories

We have developed a web interface for a historian-expert which for each category recommends the list of typically confused categories. The expert had to review each category and decide: keep a category as it is, merge it with another category or drop the category and relabel the related documents. After reviewing manually the list of categories we obtained 88 final categories.

In addition, we evaluate the agreement between human annotators. We consider the inter-annotator agreement in category assignment as a level of performance that may be achieved by automatic documents classifiers. We randomly selected 2000 labelled notary acts and asked another human annotator to assign a category after removing the label. Then we evaluated the pairwise agreement between annotators using *Cohen's kappa coefficient*. According to the weighted average kappa coefficient the annotators agree on 88.49%. The disagreement occurs because there are no clear borders between some categories.

## 4 Two Level Classification

We are interested in obtaining accurate results as well as predicting rare categories in the collection of documents $\mathcal{D}$. The prediction of frequent categories will allow us to get high performance results, but in many cases the smaller categories will be confused with the larger ones. Therefore we design an approach that takes into account the category frequency information. We introduce the following definitions:

**Definition 1.** The support of a category in a set of documents is its proportional size in the set.

**Definition 2.** The category $c \in \mathcal{C}$ is *frequent* if $sup(c)$ is above a minimum defined threshold $min\_sup$, otherwise $c$ is *non-frequent*:

$$sup(c) > min\_sup \tag{1}$$

At the first level, all infrequent categories are joined to form one cluster with the smallest categories, whereas the frequent ones make up their own cluster. The minimum support can be learnt during a training phase. We used 2%. The output of this level is a set of cluster-labels $\{f_1, ..., f_n\}$ associated with each document $d \in D$ and the set of clusters $\mathcal{F}$.

At the second level we incorporate the clustering results into a prediction model and the TC process. This idea is described in Algorithm 1.

**Algorithm 1.** Building prediction model and TC classification

**Input:** Training set $\mathcal{D} = \{d_1, ..., d_n\}$ with category-labels $\{c_1, ..., c_k\}$ and cluster-labels $\{f_1, ..., f_n\}$. Test set $\mathcal{T} = \{t_1, ..., t_h\}$. Set of categories $\mathcal{C} = \{c_1, ..., c_k\}$ and set of clusters $\mathcal{F}$. Learning algorithm of the prediction model $\mathcal{L}$

**Output:** Predicted labels $\mathcal{N}$ for all test instances $\mathcal{T}$

1. $\mathcal{N} \leftarrow \emptyset$
2. $\mathcal{M} \leftarrow TrainModel(\mathcal{D}, \mathcal{F}, \mathcal{L})$ # Learn model on cluster labels
3. $\mathcal{N}^* \leftarrow Classify(\mathcal{T}, \mathcal{M})$ # Classify test data with cluster-labels
4. **for** each cluster $f_i$ in $\mathcal{F}$ **do**
5. $\quad \mathcal{D}_i \in \mathcal{D}, \mathcal{T}_i \in \mathcal{T}, \mathcal{C}_i \in \mathcal{C}$ # Associate data with the cluster
6. $\quad \mathcal{M}_i \leftarrow TrainModel(\mathcal{D}_i, \mathcal{C}_i, \mathcal{L})$ # Learn model on category labels
7. $\quad \mathcal{N}_i \leftarrow Classify(\mathcal{T}_i, \mathcal{M}_i)$ # Classify data with final categories
8. $\quad \mathcal{N} \leftarrow \mathcal{N} \cup \mathcal{N}_i$
9. **end for**
10. **return** $\mathcal{N}$

## 5 Experiments and Results

We conducted experiments on the annotated datasets described in Section 2. We have three sets of experiments. In order to assess the performance of our results, we apply 10-fold cross-validation. Due to lack of space, a more detailed and graphical view of the experiments is available on the web[6]

**Experiment 1: TC Results before Category Resolution.** Table 2 presents the overall accuracy for each experimental setup before category resolution (i. e. with 455 categories). The best results were achieved with an SVM classifier using the complete lexical vocabulary as a features without stemming procedure and named entity elimination. We expected that the elimination of person names and locations would affect the accuracy of the classifier positively, but from the results we see the opposite: there is a small correlation between locations and person names and type of notarial acts. However, despite the

[6] `http://wwwis.win.tue.nl/~amontes/ecir2015/results.html`

**Table 2.** Performance accuracy in the experiment 1 before category resolution

| Model | Feature | Stemming | PIE | all features | chi-sq. | lsa | POS |
|---|---|---|---|---|---|---|---|
| SVM, lin. kernel | tf-idf | ✗ | ✗ | **86.84** | 84.70 | 84.03 | 86.32 |
| SVM, lin. kernel | tf-idf | ✗ | ✓ | 85.67 | 84.05 | 83.89 | 84.52 |
| SVM, lin. kernel | tf-idf | ✓ | ✗ | 86.55 | 85.22 | 84.02 | 85.11 |
| SVM, lin. kernel | tf-idf | ✓ | ✓ | 86.38 | 84.45 | 83.85 | 84.52 |

promising overall results 307 categories are completely ignored by the classifier. Therefore, we performed the category resolution described in Section 3.

**Experiment 2: TC Techniques after Category Resolution.** Table 3 presents the accuracy results for each experimental setup after category resolution. The best results again are achieved by applying a SVM classifier and using a complete sparse lexical vocabulary as feature vector without named entity elimination. The classifier in this case is not sensitive to the stemming procedure. In this experiment we achieved a maximum accuracy of 87.79% which is 0.95% higher than before. The number of categories with an absolute zero f-score is reduced to 17. That can be explained by the very few examples in each category.

**Table 3.** Performance accuracy in the experiment 2 after category resolution

| Model | Feature | Stemming | PIE | all features | chi-sq. | lsa | POS |
|---|---|---|---|---|---|---|---|
| SVM, lin. kernel | tf-idf | ✗ | ✗ | **87.79** | 86.56 | 84.86 | 87.40 |
| SVM, lin. kernel | tf-idf | ✗ | ✓ | 86.38 | 85.50 | 84.95 | 85.65 |
| SVM, lin. kernel | tf-idf | ✓ | ✗ | **87.79** | 86.80 | 85.02 | 87.42 |
| SVM, lin. kernel | tf-idf | ✓ | ✓ | 86.25 | 85.53 | 84.93 | 85.65 |

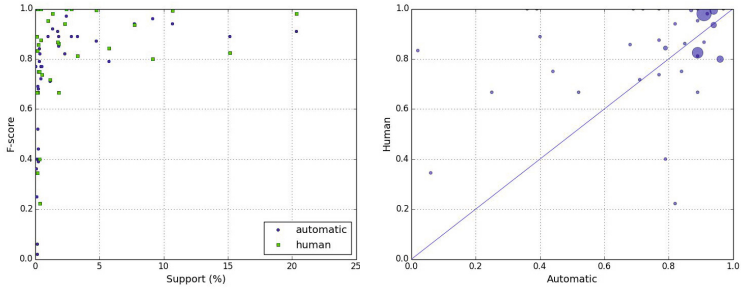
**Experiment 3: TC Using Two Level Classification.** Table 4 presents the accuracy results of the proposed framework for each experimental setup. The maximum accuracy is increased up to 88.08% which is 0.3% higher than before. There is also a slight improvement in the number of unidentified categories, it is reduced to 14 compared to 17 unidentified categories in the previous experiment. Still the very few examples in rare categories does not allow to the classifier to recognise them all. Nevertheless the proposed simple clustering technique as a framework to the overall classification process shows promising results.

**Table 4.** Performance accuracy in the experiment 3 using two level classification

| Model | Feature | Stemming | PIE | all features | chi-sq. | lsa | POS |
|---|---|---|---|---|---|---|---|
| SVM, lin. kernel | tf-idf | ✗ | ✗ | **88.08** | 87.50 | 85.60 | 87.51 |
| SVM, lin. kernel | tf-idf | ✗ | ✓ | 86.51 | 85.93 | 85.42 | 85.77 |
| SVM, lin. kernel | tf-idf | ✓ | ✗ | 88.07 | 87.60 | 85.73 | 87.52 |
| SVM, lin. kernel | tf-idf | ✓ | ✓ | 86.39 | 85.91 | 85.52 | 85.77 |

**Comparative Evaluation.** We performed a comparative analysis of our two-level classification algorithm versus human agreement (see Fig. 2). Since we have two annotators, we consider the labelling results from the $1^{st}$ annotator as the ground truth and evaluate the results of the $2^{nd}$ annotator. In Fig. 2a, we compare the results for each category for the manual and the automatic evaluation method. The small categories are recognised much better by people, while for larger categories the results are comparable. Fig. 2b shows that the

performance of humans for most of the categories correlates with the automatic classification. However there is a number of categories where humans significantly outperforms our algorithm. Those categories have a very small support value.




**(a)** F-score vs support per category **(b)** F-score for human vs automatic TC

**Fig. 2.** Evaluation of f-score for individual categories for automatic TC and human

## 6 Conclusions

In this paper we described a framework for dealing with noisy labels. We examined existing text classification algorithms, studied the influence of lexical information and analyzed a number of feature selection methods. Then we created a two level classification approach that slightly improved the results, achieving a performance close to the inter-annotator agreement. The developed methods can be applied for classification of narrative data in different domains.

## References

1. Aggarwal, C.C., Zhai, C.: A survey of text classification algorithms. In: Mining Text Data, pp. 163–222. Springer (2012)
2. Van den Bosch, A., Busser, B., Canisius, S., Daelemans, W.: An efficient memory-based morphosyntactic tagger and parser for dutch. In: CLIN, pp. 99–114 (2007)
3. Constantopoulos, P., Doerr, M., Theodoridou, M., Tzobanakis, M.: Historical documents as monuments and as sources. In: Proceedings of Computer Applications and Quantitative Methods in Archaeology Conference (2002)
4. Iglesias, J.A., Tiemblo, A., Ledezma, A.I., Sanchis, A.: News mining using evolving fuzzy systems. In: Corchado, E., Lozano, J.A., Quintián, H., Yin, H. (eds.) IDEAL 2014. LNCS, vol. 8669, pp. 327–335. Springer, Heidelberg (2014)
5. Ikonomakis, M., Kotsiantis, S., Tampakas, V.: Text classification using machine learning techniques (2005)
6. Leong, C.K., Lee, Y.H., Mak, W.K.: Mining sentiments in sms texts for teaching evaluation. Expert Systems with Applications 39(3), 2584–2589 (2012)
7. Nguyen, D., Trieschnigg, D., Theune, M.: Folktale classification using learning to rank. In: Serdyukov, P., Braslavski, P., Kuznetsov, S.O., Kamps, J., Rüger, S., Agichtein, E., Segalovich, I., Yilmaz, E. (eds.) ECIR 2013. LNCS, vol. 7814, pp. 195–206. Springer, Heidelberg (2013)
8. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. 34(1), 1–47 (2002)