**Machine Learning on the Classification of Economic Activities**

Thivyesh Ahilathasan, Statistics Norway, [Thivyesh.Ahilathasan@ssb.no](mailto:Thivyesh.Ahilathasan@ssb.no)

Kristin Foldal Haugen, Statistics Norway, [Kristin.Foldal.Haugen@ssb.no](mailto:Kristin.Foldal.Haugen@ssb.no)

Magnar Lillegård, Statistics Norway, [Magnar.Lillegard@ssb.no](mailto:Magnar.Lillegard@ssb.no)

Tatsiana Pekarskaya, Statistics Norway, [Tatsiana.Pekarskaya@ssb.no](mailto:Tatsiana.Pekarskaya@ssb.no)

**Abstract:**

*Today, an executive officer at the Central Coordinating Register for Legal Entities classifies the entities according to the activity the entities carry out. This classification is a time-consuming process and often leads to the wrong result. By these reasons a new system, which would provide automation of the classification process, is required. Statistics Norway together with other agencies in Norway has an ongoing project on developing a new system for determining standard industrial classification (SIC) codes. The new system will let the entities themselves classify their activities with the help of a trained model. In the paper, several models are presented, being developed with the help of machine learning techniques to suggest to entities SIC codes based on the companies’ business descriptions. The models tested are fastText, CNN and a two-level modelling. Additionally, there are outlighted ways to handle an imbalanced dataset, which contains a lot of mistakes.*

***Keywords:*** *classification of SIC, machine learning, text classification, quality improvemen*t

1. **Introduction**
   1. *Description of NACE and SIC codes*

NACE is the acronym for the French title ‘Nomenclature statistique des activités économiques dans la Communauté européenne’ meaning the Statistical classification of economic activities in the European Communities. “NACE provides the framework for collecting and presenting a large range of statistical data according to economic activity in the fields of economic statistics (e.g. production, employment, national accounts) and in other statistical domains. Statistics produced on the basis of NACE are comparable at European and, in general, at world level. The use of NACE is mandatory within the European Statistical System” [(*NACE: Statistical Classification of Economic Activities*, 2008)](https://paperpile.com/c/4rNQMn/4OdLU).

“The main purpose of the standard is to provide rules and guidelines for industrial classification…. To achieve uniform classification, the codings are linked to Statistics Norway's Business register and the Central Coordinating Register of Legal Entities. Industrial classification entails grouping together homogenous activities insofar as possible, i.e. classifying production units (enterprises, local units (LU) etc.) according to the economic activity in which they are engaged. In this context, the term "activity" refers to a process in which different production factors (resources such as raw materials, capital and labour) are combined, leading to the creation of specific goods or services. An activity is characterized by an input of products (goods and services), a production process, and an output of products. Activities are determined by reference to a specific level of SIC2007” (Statistics Norway, 2007).

“An economic activity takes place when resources such as capital goods, labour, manufacturing techniques or intermediary products are combined to produce specific goods or services. Thus, economic activity is characterized by input of resources, a production process and an output of products (goods or services)” [(*NACE: Statistical Classification of Economic Activities*, 2008)](https://paperpile.com/c/4rNQMn/4OdLU).

The standard of industrial classification (SIC) holds a five-level hierarchical structure. SIC has the first four levels defined analogically to NACE code and one additional level which is defined nationally. The designations of *SIC code* levels are described in Table 1.

**Table 1. The hierarchical structure of NACE and SIC**

|  |  |  |  |
| --- | --- | --- | --- |
| *Level* | *Code example* | *Level is defined by* | *Code Designation Number* |
| Section | C | NACE | 21 |
| Division | 10 | NACE | 87 |
| Group | 10.2 | NACE | 270 |
| Class | 10.20 | NACE | 613 |
| Subclass | 10.201 | SIC | 820 (299 of them are national codes) |

Source: Statistics Norway

* 1. *A traditional way of economic activities classification*

In the Central Coordinating Register for Legal Entities (CCRE) both the legal unit and its local kind of activity units are registered. All these units further we will further refer to as *entities*.

An example of registration in CCRE will be provided for legal units. When in Norway legal units register in the CCRE, they do this primarily to be assigned with an organization number, which is playing the role of an ID. The legal units need the ID for their registration in tax authorities, to get a bank account and other. They fill in a web form with the necessary information. One of the information elements in the form is a description of a type of activity they are planning to perform (*economic activities description*). Not all respondents are aware that afterwards, their description of economic activity results in a SIC code in a register. Therefore, the quality of the information can be poor. Their prime need is to receive the organization number and be legally registered in the CCRE. They do not focus on the quality of the SIC codes in the register.

An executive officer at the CCRE classifies the entities according to description of economic activities the legal units have reported. The description of economic activities provided by the legal unit can be inaccurate and has poor quality. If this is the case it results in either a request to the company asking for more details or in an unaided interpretation by the executive officer. In the last case, depending on the executive officer, the classification can be carried out differently, caused by their experience and knowledge. By these reasons, the classification can often lead to a wrong result.

Classification of economic activities to a corresponding SIC code is a time-consuming and ineffective process. To reduce time caused by the classification work, one should facilitate a better dialogue with legal units. From the web form used today, the legal unit does not know that their description of activities leads to a SIC code. In a new user-interface, this could be solved. To deal with all the problems described above, a new solution is needed.

1. **Problem statement**

Statistics Norway together with other agencies in Norway has an ongoing project on developing a new system for determining SIC codes. Today, an executive officer at the CCRE classifies the entities according to the reported activities they carry out. The new system will let the entities themselves classify their activities proposing them relevant SIC codes. SIC codes are planned to be generated by a model trained with machine learning(ML) techniques, based on descriptions of economic activities provided by entities.

Further, in the paper, we give a description of the results we got, paying attention to data available for us, problems caused by them, and the choice of model. In the end, we discuss the perspective of special solutions, which could improve the quality if the final solution.

1. **Problem solution**

The stated problem in ML is defined as a *multi-label classification problem*. It is traditionally solved by training a model, which associates an input (economic activities description) with a label (a SIC code) from a set of labels . For a multi-label classification problem, size of is more than two ().

* 1. *Available data and difficulties they bring*

Available data are:

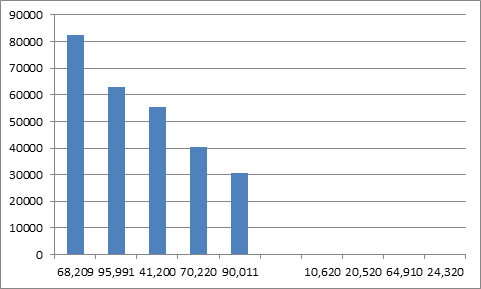
1. Descriptions of economic activities . This is the main source of information in our project.
2. Currently defined SIC codes for each description .
3. ‘Official’ descriptions of codes and keywords from the SIC system version 2007 [(*NACE: Statistical Classification of Economic Activities*, 2008)](https://paperpile.com/c/4rNQMn/4OdLU) .

The analysis of the available data showed the existence of some data obstacles, which influence the model performance.

1. For some activities’ description, there are defined more than one SIC code. When filling out the web form for registration in the CCRE, the legal unit can specify many different activities. They are informed that it is important to list their principal activity first and add secondary activities after. There is only in some cases that the legal unit receives more than one SIC code, depending on the statistical needs. The result of this is sometimes a text containing information on *many* activities, but in the dataset, it only corresponds to *one* SIC principal code. For instance, if the legal unit describes both their activity as *construction activities (43.1)* and adds *management consultancy activities (70.2)* this record is only given the principal code 43.110*.*
2. Legal units which are a part of an enterprise group have some special rules when it comes to definition of their SIC code. The legal units which only do business within the enterprise group will get the same code as their closest parent company. This means that their business description doesn’t fit with a given to them SIC code. Further on, parental companies that only handle the administrative parts of the business for the enterprise group, are not coded for their administrative work, but coded after one of their daughter companies. The reason is that the whole enterprise group gets the same code as its parent company.
3. As it was mentioned in Section 1, because of the human factor, mistakes are often made when classifying an activity description. A Manual correction of the classification is time-consuming and can still lead to mistakes. Another problem is that executive officers at Statistics Norway have changed SIC codes when new information through surveys or direct contact with the legal units tells us that the SIC code is either wrong or outdated. This happens without the descriptions being changed. By these reasons, in the available data, there exist mistakes in the associations between with .
4. It’s a fact that some economic activities are very rare and other very popular. This causes different frequencies of descriptions available for each SIC code . Such a problem in ML is known as *imbalance*. A model basing on imbalanced dataset will be well trained in predicting frequent codes (the majority classes). Simultaneously, the model will be less trained in predicting those codes which are not so frequent (the minority classes).

From Figure 1 and Table 2 it is possible to see that the number of activity descriptions for each code varies from 1 (there are 7 codes like this) to 82,552 (code 68.209).

**Figure 1. Imbalanced dataset**



Source: Statistics Norway

**Table 2. Imbalanced dataset**



Source: Statistics Norway

1. Some activities descriptions can be suited to several SIC codes, if not looking at Exclude notations in official descriptions. Or they can have keywords, which are important for the decision whether it is one NACE or another.

To eliminate the first obstacle, only entities with *one* SIC code were included in the input data. We removed the records where the legal units had been classified with both principal and secondary activities. These records contain more than one SIC code.

Ancillary units were removed, too, because these are classified with the same SIC code as the entities they are helping, and not within the activity they actually are performing.

As far as the third obstacles is concerned, it could be solved by codes correction or removing those associations of and , which are wrong. These goals could be reached by providing manual correction/removing or by training one more module, which could identify wrong associations. Both processes require high time consumption and do not provide 100 percent accuracy.

Apostolova and Kreek (2018) address a question of how well performance metrics computed on dirty, historical data reflect the performance on the intended future ML input. The results of the work have shown that the accuracy on a clean dataset significantly outperforms the accuracy measured on the dirty training sets via cross-validation. This suggests that traditional accuracy measures on dirty training datasets are typically over-pessimistic, which is positive.

However, the accuracy will of course increase if the number of errors in the dataset is reduced. The clean lab Python package, [(cgnorthcutt, n.d.)](https://paperpile.com/c/4rNQMn/kgNo) was designed to deal with the problem of noisy labels in multi-label text classification. The usage of the package will be considered for testing in further work and is not described in the paper.

An imbalanced problem can be solved by making a more balanced set or by making weighting of labels.

A way of changing the number of examples can be reached by at least two solutions:

1) undersampling of the majority classes,

2) oversampling the minority classes.

Package imbalanced-learn in Python is built for solving imbalanced dataset problem. However, application of most of the methods from the package requires a vector representation of inputs. Only random undersampling and oversampling can be applied for activities descriptions with text representation (a case of fastText). If other methods are wanted to be applied for fastText, additional representation for input is needed.

For the generation of synthetic data (oversampling) as a solution can be useful a creation of examples from codes’ descriptions or replacement of some word with keywords.

Modifying the dataset with resampling-like methods is changing the reality. One must be careful and have in mind what it means for the output results of his classifier. See [(Rocca, 2019)](https://paperpile.com/c/4rNQMn/txn9) for more details.

Weighting of models can be applied by using such model as, for example, Random Forest. The label of the majority class should get a lower weight than the one, which is of minority. In some Python packages the weighting is already implemented and can be easily used. Here it is important to choose a right weighting function.

The last problem of similar descriptions can be weakened adding artificial activities descriptions with keywords and using the ‘official’ descriptions of codes from the SIC system.

* 1. *Choice of model*

There is a list of models, which handle multiple text classification problems. Among them there are Naive Bayes, Random Forest, Convolutional Neural Network (CNN), fastText and others. All of them have their advantages and disadvantages.

We decided to implement several models to compare results and to choose the one with the highest performance. The models which were chosen for development are CNN, fastText, and a two-level model, which is a combination of two fastText or fastText and Random Forest.

We have chosen to use fastText, because it provides possibilities to train a model extremely fast and is easy for implementation. Based on the results of fastText training we end up with an idea to use a two-level model, combining it with other models.

Neutral Networks earned fame for being high-performing models. Convolutional Neural Network (CNN) lately got a high popularity not only for image classification, but also for text. By this reasons CNN was chosen for development, too.

In this subsection descriptions of models and representations of input data are provided, with pre-processing processes required for each of them.

1. *Data pre-processing*

Depending on the choice of model, input data possess different representation. Not depending on the input representation, input data require pre-processing before training a model.

In our work we test the next data pre-processing methods:

* Modification of all descriptions to lower case,
* Removal of punctuation and digits,
* Removal of stop words in sentences,
* Stemming of words.

In further work will be considered for testing decoding of met abbreviations.

1. *fastText*

A library for a fast classification of text, called fastText, was developed by a Facebook team and released in 2015.

For a given sentence the model first makes a vector representation of each word or a combination of words (n-gram) in a space. A usage of n-grams can be beneficial for taking into consideration words order. Those words and n-grams, which appear to have the smallest distance between each other in the vector space, are counted to be similar. After the vectorisation of words and n-grams is completed, a classifier uses a softmax function to calculate a probability of all labels’ associations for each description [(*FastText Tutorial - How to Classify Text with FastText*, 2017)](https://paperpile.com/c/4rNQMn/APgw).

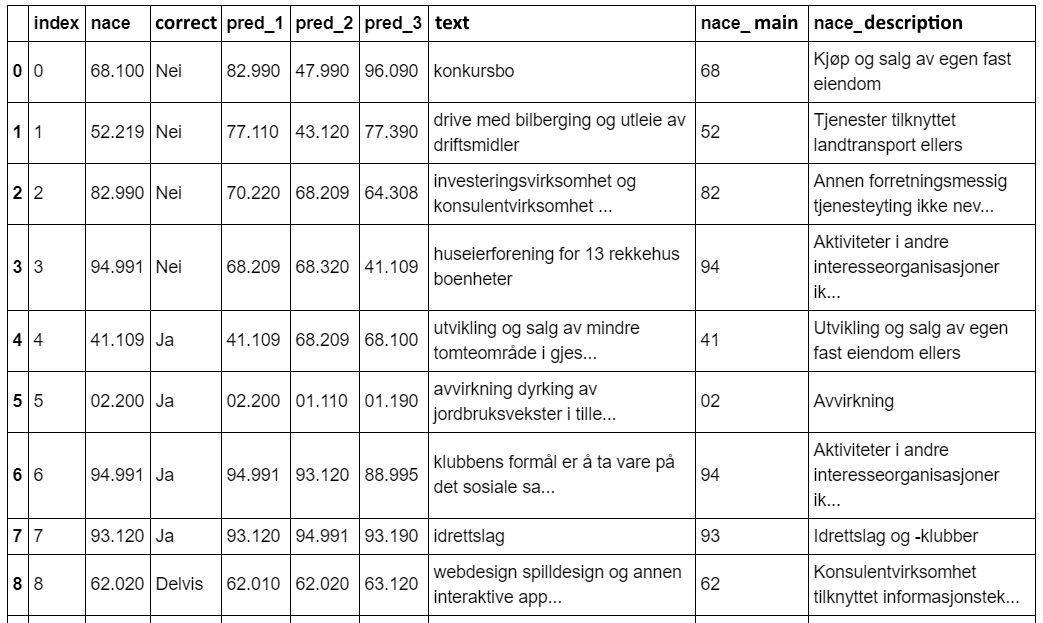
A label, having the highest association probability, is counted to be predicted and based on it is calculated accuracy and f1 value of a prediction model.

For model validation, we are looking additionally at an accuracy of partly correct predictions. It is assumed that a label is predicted partly correct, if it was met among five first predicted labels. If at least one of them gave a right prediction, we can assume that the model made a correct prediction. An example of partly correct prediction can be seen in Figure 2. In the figure a description with index eight, was predicted partly correct, getting a correct result in the second prediction. For the rest of the indexes the results are either correct, or wrong.

One improvement for fastText is a pretrained vector which is tested on Wikipedia dataset on *Bokmål* and *Nynorsk* (the two Norwegian written languages). Pretrained vectors often provide better word vectorization than using just input dataset, because it has basically more words.

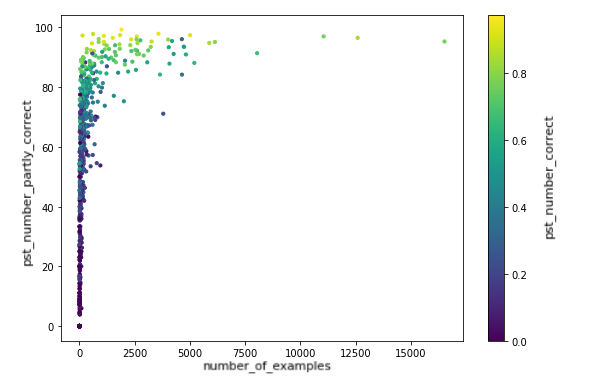
Testing of fastText has shown that the model does not handle imbalanced problems good enough. In Figure 3 and Tables 3, 4 we can see that those codes which had more examples in the group, have shown high results in comparison with small groups. The last have a high variation in the number of correct/partly correct examples. The results show that there should be put an attention to the imbalance problem.

**Figure 2. An example with correct, partly correct and wrong predictions**



Source: NAV, Paul Bencze

**Figure 3. A test of a fastText model**



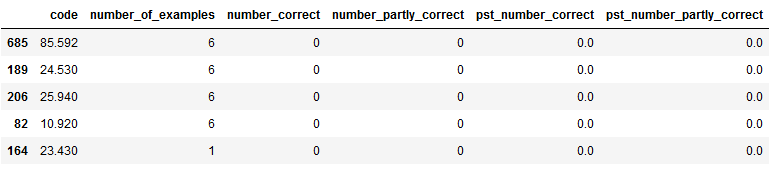
Source: Statistics Norway, based on code of NAV, Paul Bencze

**Table 3. A test of a fastText model, showing five groups with the highest percent of partly correct predictions**



Source: Statistics Norway

**Table 4. A test of a fastText model, showing five groups with the worst percent of partly correct predictions**



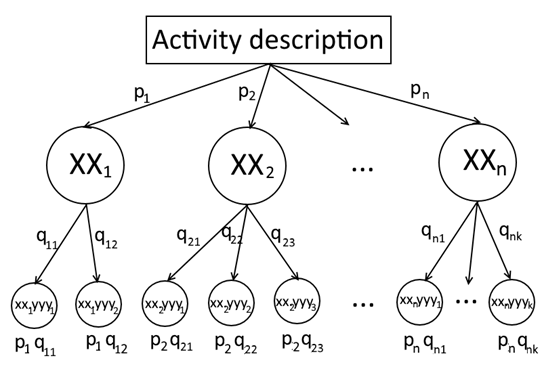
Source: Statistics Norway

1. *Two-level modelling*

Results of fastText application have shown that the model cannot perfectly handle an imbalanced dataset. That is why we decided to build a two-level model, which would help to improve the imbalance problem. A multi-level model can be beneficial from the side that keywords or descriptions of SIC codes can be used for different levels of the model. For instance, for prediction of two first digits, a Division description of the SIC code could be used, and for five digits – description of Subclasses.

Multilevel classification models were described inWenhu Yu et al. (2018). Analogically to the work we train a two-level model (Figure 4). At the first level Random Forest or fastText is used, which provide predictions of two first digits of SIC. At the second level a model from fastText library is applied, which provides predictions of five digits.

**Figure 4. Two-level text classification model**



Source: Statistics Norway

After application of both models, one gets two sets of probabilities:

1. probabilities of two first digits of SIC ();

2. probabilities of five digits of SIC (.

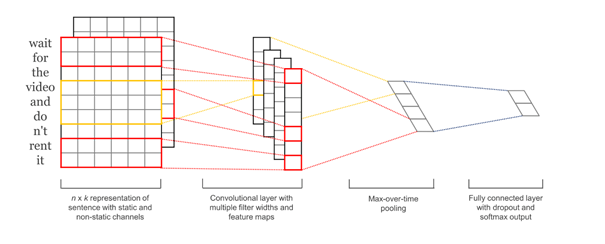
To define the most probable codes, there is calculated a combined probability of labels from both models ().

Random Forest is known to handle imbalanced problems, using weighted classes. Thus, it could improve results of independent fastText, by assigning probabilities to divisions’ groups (two first digits).

1. *CNN*

One kind of neural networks tested to solve the problem is convolutional neural network (CNN). CNNs have revolutionized image classification and computer vision by being able to extract features from images. The ability to extract features has also proven to be very useful when it comes to sequence processing and text classification [(“Introduction | ML Universal Guides | Google Developers,” n.d.; Kim, 2014)](https://paperpile.com/c/4rNQMn/QceC+AVhr). One could think of CNN as a specialized neural network that can detect specific patterns. The way CNN works is that it has hidden layers as shown in Figure 5. These are called convolutional layers which consist of filters being used to detect specific features. These convolutional layers can detect edges, corners and other types of textures in an image by using these filters which are moved across the image. The network detects more complex patterns with each convolutional layer.

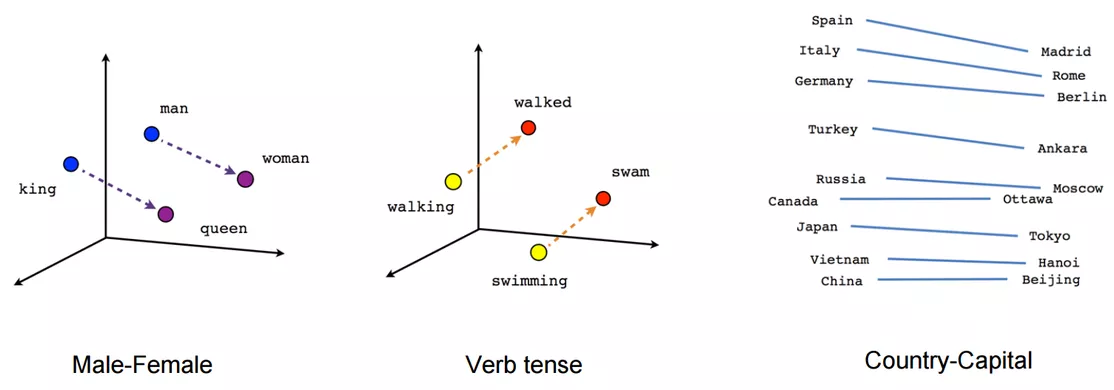
**Figure 5. A visualisation of a convolutional neural network used for text classification**



Source: [(Kim, 2014)](https://paperpile.com/c/4rNQMn/QceC)

The same goes for sequence processing and text classification [(Python, 2018)](https://paperpile.com/c/4rNQMn/s5wWT). But for the model to be able to detect features in the sentence we will have to pre-process the input data first. As a step in the pre-processing described earlier, we use word embedding to represent each word as a vector. Word embedding is the process where words with similar meaning are represented similarly. This method is widely used in natural language processing (NLP). Word embeddings are words represented as dense word vectors. By representing words as word embeddings, one collects more information into fewer dimensions. This is crucial because most neural networks don't work well with high-dimensional sparse vectors but work optimally with lower dimensional dense vectors. By using low-dimensional dense word vectors, we can map features that provide similar clues through a representation that is able to capture these similarities. Hence, the aim of word embeddings is to map the semantic meaning of the words into a geometric space. This geometric space is also called the embedding space (Figure 6). If the word embedding captures the similarities and the differences between the words well, one could perform arithmetic operations such as: King - Man + Woman = Queen.

**Figure 6. Visualizing the embedding space for different applications.**



Source: [(NSS, 2017)](https://paperpile.com/c/4rNQMn/5SEz)

After pre-processing, CNN will be able to map different features by working its way through the word embeddings. Technically, the model, depending on the size of the filter, slides through the word embedding and extracts features by convolution. The word embeddings enables the model to map features in an efficient manner, because of the low dimensional dense word vectors. Combined with an activation function and a bias term it maps the features. The bias term, further on, the model pools and concatenate the feature vectors to a fixed length vector. This fixed length vector is then fed into a fully connected softmax layer to perform the classification shown in Figure 5 [(Britz, 2015)](https://paperpile.com/c/4rNQMn/rfxqK).

1. **Experiments**
   1. *Tests setup and Environment for testing*

For problem solution the Python 3 language was chosen. It contains a significant number of packages handling test classification problems. Python 3 is widely used for ML because of the availability of such libraries as Keras, NumPy, SciPy, ScikitLearn, PyBrain etc. Further on, Python 3 is easier to learn from scratch as opposed to Java, C/C++. We use a platform called Jupyter notebook, where Python 3 can be used together with other languages.

The final dataset contains around 1,5 million descriptions of activities and 821 five digits’ labels. For labels are used 820 from official description and one additional 00.000. For testing the results of the models with different pre-processing operations, equivalent splits for train and test datasets were used with proportion 80 and 20 per cent respectively. In the splits was considered imbalanced problem, meaning that proportion 80/20 was applied for each label.

* 1. *Results*

Results of fastText testing can be seen in Tables 5. The table shows results for correct, partly correct (see Section 3.2.2) predictions and time spent for model training.

The best result of for fastText model were reached with such pre-processing operations as:

1. lower case
2. partly removed punctuation and digits
3. keywords
4. bokmål pretrained vector
5. extended stopwords list.

For the same model fully removed punctuation and digits decreased the accuracy. Nynorsk pretrained vector provided worse results than bokmål. Stemming did not show significant improvements.

For two-level modelling for stages where fastText is used were applied the same pre-processing operations.

**Table 5. Results of fastText**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Accuracy** | **f1** | **Partly correct accuracy** | **Partly correct f1** | **Time for training** |
| **0.6082** | **0.2552** | **0.8572** | **0.4800** | **527.19s** |

Source: Statistics Norway

As it can be seen from Table 6 the results of fastText were not significantly improved, while required more time consumption, pointing that a simple application of fastText is preferable.

**Table 6. Results of two-level model (fastText + fastText)**

|  |  |  |
| --- | --- | --- |
| **Accuracy** | **f1** | **Time for training** |
| **0.6090** | **0.2618** | **943.67s** |

Source: Statistics Norway

In order two-level modelling gives better results than ordinary model, classification of activities to two first digits of SIC should have higher performance than provided in Table 5. Application of RandomForest for two first digits provides us lower performance. That is why the results of this two-level model are not shown in the paper.

The CNN is currently at the stage of hypertuning, which means that different parameters are tested and tweaked to achieve a higher accuracy. The loss function and the learning rate scheduler which are used are respectively categorical cross-entropy [(“Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names,” n.d.)](https://paperpile.com/c/4rNQMn/WuN0) and adam (Kingma & Ba, 2014). CNN compared to fastText is more time consuming. If the performance of CNN turns out to be not significantly higher than of fastText, then the last would be the clear choice.

Currently fastText is the best solution, providing the highest performance for a significantly fast time. However, a satisfying level of accuracy and f1 is not reached yet, requiring further improvement of the model.

1. **Improvements of the final solution**

An ideal prediction can never be reached. However, steps for getting a satisfying performance should be executed, by making improvements of the current solution.

## Dealing with incorrect classification

Some manual work has a place and must be estimated. In the feasibility studies from 2017/2018 it was specified that the descriptions given by entities in a new user interface need to be stored to be able to control the chosen codes afterwards.

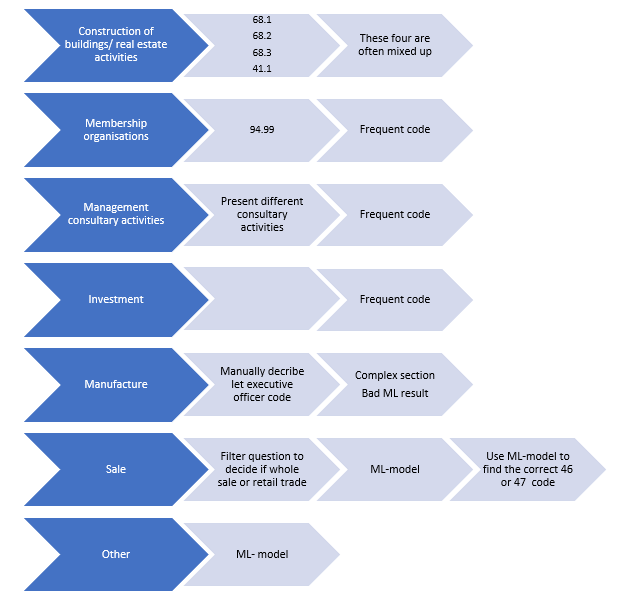
## Introducing special solutions

To make the process of SIC determination in the new system better, it is considered to add special solutions to the final user interface. A solution where filter questions divide/sort the entities correctly from the start could decrease number of wrong classifications. Seeking advice from form methodologists could be essential here. Additional classification of SIC codes with ML for each additional class could give better results of the final solution. The additional classification should be done at the first stages before an entity provides an activity description. Such solution would be the most beneficial defining additional classes for those SIC codes which are known to be the most problematic. An example can be seen in Figure 7.

There are different reasons why we believe it is preferable to make a special solution:

* Presence of SIC codes which are often mixed up with each other and could be identified with the help of filter questions. For example, *Real estate activities.* History shows that classifying correctly into the groups 68.1, 68.2, 68.3 and the related group 41.1 is difficult and many mistakes are done.
* Presence of too frequent SIC codes. The most used SIC codes could be presented at the beginning of additional classes list, making it easy for entities finding their right codes.
* Complexity of some official descriptions for understanding. For example, *Section C.* There is a need of possible special treatment for classifying into this section, because of very complex descriptions and a few records in the training dataset.
* Similarity of some codes. For example, *Section G*. To be able for the entities easier to identify whether their activity is within wholesale 46 or retail trade 47, a layer with extra questions could be a good solution (Is your customer an individual person? Is your customer a legal entity?).

**Figure 7. An example of additional classification**



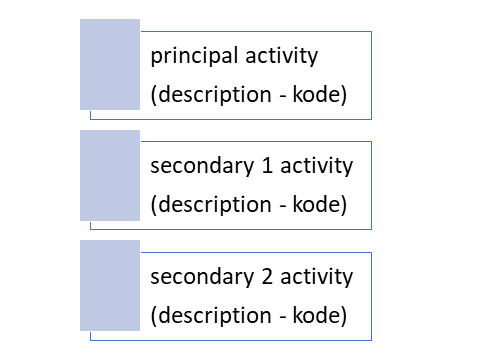
Source: Statistics Norway

A solution like this leads to higher expenses in production but could increase the quality of the final solution. Gaming away a bit in friendliness of user-interface, by requiring from an end user making more steps.

* 1. *Secondary activity*

Today the descriptions provided by entities can contain a mixture of different activities correspondent to different codes. So, one should consider creating a user interface where the entities are asked to describe their principal activity first, choosing a correspondent SIC code. Afterwards the entity can be offered to add secondary activities, going through the same procedure for code definition. In this way principal and secondary activity descriptions will be stored separately from each other, providing a right association between and and giving possibilities for improving final solution.

**Figure 8. An example for principal and secondary activities classification**



Source: Statistics Norway

1. **Conclusion**

The developed models have shown that fastText provided the best results, however not reaching the satisfying performance. By these reasons further the work with model improvement will be done. The CNN and two-level modelling are still considered for development, however showing a high consumption of training and pre-processing time. Which model to use for the final solution is not concluded yet.

In the ongoing project the decision, weather we should integrate the ML-model in a registration form, is not made yet. The executive officers have already started to test the application of classification model in their work, but the registration form used by the entities has not been changed yet.

1. **References**

[Britz, D. (2015, December 11). Implementing a CNN for Text Classification in TensorFlow. Retrieved May 14, 2019, from](http://paperpile.com/b/4rNQMn/rfxqK) <http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/>

[cgnorthcutt. (n.d.). cgnorthcutt/cleanlab. Retrieved May 15, 2019, from](http://paperpile.com/b/4rNQMn/kgNo) <https://github.com/cgnorthcutt/cleanlab>

[*FastText Tutorial - How to Classify Text with FastText*. (2017). Retrieved from](http://paperpile.com/b/4rNQMn/APgw) <https://www.youtube.com/watch?v=4l_At3oalzk>

[Introduction | ML Universal Guides | Google Developers. (n.d.). Retrieved May 15, 2019, from](http://paperpile.com/b/4rNQMn/AVhr) <https://developers.google.com/machine-learning/guides/text-classification/>

[Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. https://doi.org/](http://paperpile.com/b/4rNQMn/QceC)[10.3115/v1/d14-1181](http://dx.doi.org/10.3115/v1/d14-1181)

[*NACE: Statistical Classification of Economic Activities*. (2008). EUR-OP.](http://paperpile.com/b/4rNQMn/4OdLU)

[NSS. (2017, June 4). An Intuitive Understanding of Word Embeddings: From Count Vectors to Word2Vec. Retrieved May 15, 2019, from](http://paperpile.com/b/4rNQMn/5SEz) <https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/>

[Python, R. (2018, October 24). Practical Text Classification With Python and Keras – Real Python. Retrieved May 14, 2019, from](http://paperpile.com/b/4rNQMn/s5wWT) <https://realpython.com/python-keras-text-classification/>

[Rocca, B. (2019, January 27). Handling imbalanced datasets in machine learning. Retrieved May 15, 2019, from](http://paperpile.com/b/4rNQMn/txn9) <https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28>

[Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names. (n.d.). Retrieved May 15, 2019, from](http://paperpile.com/b/4rNQMn/WuN0) <https://gombru.github.io/2018/05/23/cross_entropy_loss/>

Apostolova, E. and Kreek, R.A., 2018. Training and Prediction Data Discrepancies: Challenges of Text Classification with Noisy, Historical Data. *arXiv preprint arXiv:1809.04019*.

Yu, W., Sun, Z., Liu, H., Li, Z., & Zheng, Z. Multi-level Deep Learning based E-commerce Product Categorization.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Statistics Norway (2008). Standard for Næringsgruppering. Statistisk Sentralbyrå.