Sentence Classification for Investment Rules Detection

Youness Mansar and **Sira Ferradans**
Fortia Financial Solutions
17 av. George V, Paris
France
name.surname@fortia.fr

Abstract

In the last years, compliance requirements for the banking sector have greatly augmented, making the current compliance processes difficult to maintain. Any process that allows to accelerate the identification and implementation of compliance requirements can help address this issues. The contributions of the paper are twofold: we propose a new NLP task that is the investment rule detection, and a group of methods identify them. We show that the proposed methods are highly performing and fast, thus can be deployed in production.

1 Introduction

Compliance requirements have augmented dramatically in the last years, specially in the financial sector. Investment funds are obliged by law to publish their investment strategy at a very detailed level. If the fund does not follow precisely these rules, it will be fined by the corresponding regulatory institution. According to Thomson Reuters there were regulatory changes every 12 minutes, on average per day in 2015 (Thomson Reuters, 2015). But, it takes months to implement every regulatory change, thus, any process that allows to spot regulatory changes can help accelerate this updating process. This is important since if an investment fund does not follow precisely these rules, it will be fined by the corresponding regulatory institution. In fact, in the last years, the income dedicated to fines and settlements has increased by almost 45x for the biggest EU and US banks (Kaminski and Robu, 2016).

The compliance department of Depositary banks are in charge of controlling that these rules are actually followed. In order to avoid sanctions, they define a 4-eye protocol for rule identification. This protocol consists in having two or more people read and highlight the investment rules of the prospectus of each investment fund they control. Once two people have highlighted the same prospectus, a third person introduces all the rules in the system. Identifying the rules is time consuming and tedious. This process takes days for human actors, we propose a method that takes seconds thanks to the use of machine learning. Although other methods have acknowledged the importance of having the rules isolated (Cashman et al., 2002; Beale, 2004), the current systems assume that the rules have already been identified and translated into executable code.

In this paper, we propose to detect investment rules using binary classification of sentences. In section 2, we present the state of the art in sentence classification. In section 3.1, we give all the details on the data and the posed problem. The proposed solutions are given in section 3.2 and the obtained results in section 3.3. Section 4 concludes the paper and gives future work. [1]

2 Related Works

Sentence Classification. Sentence classification is a classic research area in natural language processing. Approaches previous to 2010 focus mostly on the extraction of document meaning through representative features that would be used as input to classic machine learning algorithms, such as SVM, knn, or Naive Bayes (see (Khan et al., 2010) for a review on the topic). The rise of Deep Learning techniques impacts also the sentence classification literature, appearing methods based on CNNs. More specifically, a modification of (Collobert et al., 2011) was proposed by

[1] **Note:** There is a Patent Pending for the presented approach. It was submitted the 18 December 2017 at the EPO and has the number EP17306801

Proceedings of the First Workshop on Economics and Natural Language Processing, pages 44–48
Melbourne, Australia, July 20, 2018.

Kim (Kim, 2014), showing how a simple model together with pre-trained word representations can be highly performing. But the use of word-embeddings has been challenged for CNNs, (Johnson and Zhang, 2014, 2015) propose a semi-supervised setting that allows to learn a small text-region representation. Zhang et al. (Zhang et al., 2015) propose a CNN based directly on character representations, without explicitly encoding words. CNNs are highly dependent on the window size, (Lai et al., 2015; Visin et al., 2015) propose the use of Recurrent Convolutional Neural Networks to overcome this issue. (Guggilla et al., 2016) propose the use of LSTMs for classification of online user comments. In order to avoid problems due to lack of data, (Liu et al., 2016) propose multitask learning using LSTMs.

Word embeddings. The lack of big databases with tagged data is a common problem for Deep Learning models. Collobert *et al.* (Collobert et al., 2011) empirically proved the usefulness of using unsupervised word representations for a variety of different NLP tasks and since then, it is widely accepted that, for small and middle size databases ($<$ 10k samples), the use of word embeddings improves the final results. *Word embeddings* is the name associated to a group of language model methods that map words into a vector space. Introduced by Bengio et al. (Bengio et al., 2003), the authors proposed a statistical language model based on shallow neural networks. The goal was to predict the following word, given the previous context in the sentence, showing a major advance with respect to n-grams. Collobert *et al.* (Collobert et al., 2011) set the neural network architecture for many current approaches. Mikolov *et al.* (Mikolov et al., 2013) proposed a simplified model (`word2vec`) that allows to train on larger corpora. They also show how semantic relationships emerge from this training. Pennignton *et al.* (Pennington et al., 2014), GloVe, maintain the semantic capacity of word2vec while introducing the statistical information from latent semantic analysis (LSA) showing that they can improve in semantic and syntactic tasks.

3 Rule detection in prospectus

In this section we present the problem of *rule detection* in investment fund prospectus, and our proposal for tackling it.

3.1 The data

Investment fund prospectus are papers where the fund informs the regulatory institution and its future clients of its investment strategy, its risk management, the company structure, etc. Most of these documents are publicly available in the regulation authority web page, see for instance for French documents (AMF, 2018). The investment rules that we want to identify are very precise rules which can be of different kinds, and, in general, very different from other sentences in the same text as can be observed in Table 1.

The Gold standard database. The data used in the supervised part of the model is around 3.5k annotated sentences for each language (English and French). The sentences were split into two classes, the label 1 is used for rules and 0 is used for non rules, as shown in Table 1.

3.2 Proposed methods

In this subsection we detail the proposed algorithms. The task required multiple pre-processing steps that are used for data preparation before training or inference. The first step is to segment the document into a list of sentence then each sentence is tokenized into multiple elements based mostly on space and punctuation characters. Each token is then mapped to a unique id in order to produce a list of integer from each sentence which then will be fed to the regression model.

Word embeddings. The word vector values are initialized using the GloVe algorithm Pennignton *et al.* (Pennington et al., 2014) and then fine-tuned along with the model regression parameters during training. We used a corpus of fund prospectuses and wikipedia pages to train a domain-specific word embedding. This is justified by the fact that some words used in prospectuses are uncommon in the general use of language and thus are not included in available word vectors pre-trained on Wikipedia or common crawl alone.

3.2.1 Linear network model

The Linear network model in this case is a logistic regression applied to an un-weighted average of dense word vectors. The advantage of this model is that it is simple while it also takes advantage of the unsupervised pre-training of the word embeddings. This also means that is very fast and computationally cheap compared to other models

Sentence	Tag
The Fund will invest at least 70% of its net assets in sub investment grade corporate debt securities with a credit rating equivalent to BB+ or lower and denominated in USD.	1
The SICAV may invest in OTC markets.	1
The Company may not invest in gold, spot commodities, or real estate	1
The management fee is 0.1%	0
The asset manager JP Morgan assigns BNP Security Services as its depositary bank.	0

Table 1: Examples of sentences in the Data base.

presented here. In Figure 1, we can see the overall architecture of the model.

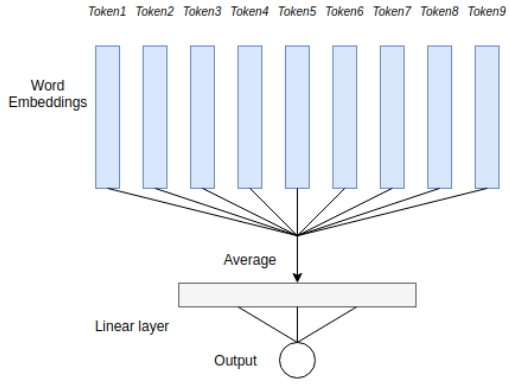

Figure 1: Linear Network architecture

3.2.2 Convolutional Neural Network

We used a CNN architecture similar to the one introduced in (Kim, 2014). It consists of the following layers:

- Convolutional Layer : Three 1-dimensional convolution layers applied in parallel to the input embedding sequence. Each convolution layer uses a different filter size {3, 4, 5} and captures sentence information at different scales (3-gram, 4-gram, 5-gram). The convolution filters learn translation-invariant representations which is useful for language because it allows for weight sharing between neurons and thus reduces significantly the number of weights compared to a fully connected layer. We use 100 filters for each layer and ReLu as a non-linearity for the convolution layers.

- Max-pooling : Applies a max operation across the sequence and returns an output that has the same size as the number of filters in each convolution layer.

- Concat Layer : Concatenates the output of each Max-pooling together.

- Linear Layer : Applies a linear mapping from the concat layer to the output.

- Sigmoid Activation : Maps the output to the [0,1] range.

In Figure 2, we can see the overall architecture of the model.

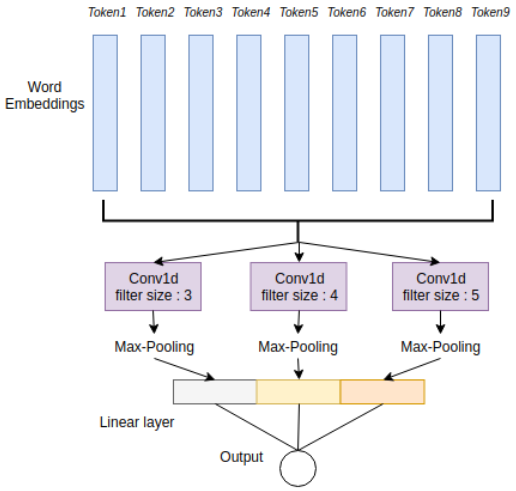

Figure 2: CNN architecture

3.2.3 Bi-directional Long-Short-Term-Memory

The Bi-LSTM model was first introduced in (Graves and Schmidhuber, 2005). Here, we used a specific model that consists of the following Layers :

- Forward LSTM : Sequential layer that is applied to the list of word embeddings from the

first token in the sentence to the last token and outputs the lstm cell state of the last token of the sentence.

- Backward LSTM : Sequential layer that is applied to the list of word embeddings from the last token in the sentence to the first token and outputs the lstm cell state of the first token of the sentence.

- Concat Layer : Concatenates the output of each LSTM layer.

- Linear Layer : Applies a linear mapping from the concat layer to the output.

- Sigmoid Activation : Maps the output to the [0,1] range.

In Figure 3, we can see the overall architecture of the model.

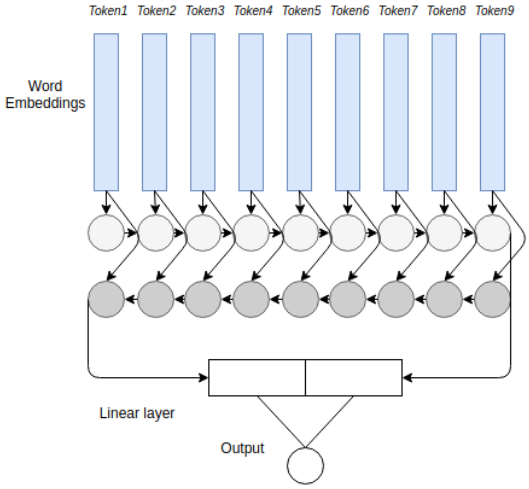

Figure 3: LSTM architecture

3.2.4 Implementation details

We used Keras (Chollet et al., 2015) with TensorFlow Backend throughout our experiments.
We use Adadelta (Zeiler, 2012) Optimizer with a learning rate of 0.001 and a batch-size of 50.
A Dropout (Srivastava et al., 2014) of 0.5 is used after the concat layer for LSTM and CNN and after the average layer for the Linear network model for regularization.
We used Binary Cross-entropy in all the models losses.

3.3 Results

We present a performance comparison of the architectures described above both in terms of accuracy/Precision/recall but also in terms of inference time as it is a also an important metric to consider when deploying a model in a production environment.

Model	Acc (std)	P (std)	R (std)
Linear	88.2(1.5)	88.2(3.3)	73.5(3.5)
CNN	**93.7**(1.0)	**90.8**(2.6)	**89.7**(3.8)
Bi-LSTM	93.3(1.1)	90.5(3.0)	88.8(2.7)

Table 2: French 10-fold Cross-validation results

Model	Acc (std)	P (std)	R (std)
Linear	87.7(3.5)	83.3(4.1)	60.8(1.4)
CNN	**94.3**(1.4)	**90.4**(4.2)	**85.8**(2.3)
Bi-LSTM	93.7(1.1)	88.8(1.9)	84.7(5.3)

Table 3: English 10-fold Cross-validation results

The convolutional model seem to yield slightly better results on average compared to the Bi-LSTM which is in line with the results presented in (Guggilla et al., 2016). Both Bi-LSTM and CNN outperform the linear network model because they take into account the order of tokens in the sentence while the linear network model does not.

Model	Time per sample (s)
Linear	$1.2e^{-4}$
CNN	$3.1e^{-4}$
Bi-LSTM	$1.8e^{-3}$

Table 4: Inference Time performance comparison

Because of its simplicity the linear network model is the fastest out of the three and the Bi-LSTM is 6 times slower than the CNN while giving worse results.

4 Conclusions and further work

We have presented a method to detect and isolate mandatory rules in regulatory documents. The objective is to automate the detection of investment rule in prospectuses using a classifier. This helps compliance experts avoid the tedious work of reading documents that are sometimes as long as 500 pages and take days to read in order to select very few sentences.

We described the frameworks used, the preprocessing steps and compared multiple classification models in terms of Accuracy/Precision/Recall and inference time. The results show that convolutional neural networks have the best trade-off between accuracy and execution time and are thus the best model for this task.

References

AMF. 2018. Geco database. http://geco.amf-france.org/Bio/rech_opcvm.aspx. Online, accessed: 2018-04-18.

N.C.L. Beale. 2004. System and method for generating compliance rules for trading systems. https://www.google.com/patents/EP0990215B1?cl=en. EP Patent 0,990,215.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research* 3(Feb):1137–1155.

D. Cashman, D. Damphousse, V. Drysdale, L. Ekhtman, C. Guerriero, K. Hebert, R. Kumar, R. Leeper, H. Levine, B. Mandel, et al. 2002. Systeme de gestion de fonds financiers et de conformite aux directives en matiere d'investissement de portefeuille. https://www.google.com/patents/EP1212711A1?cl=fr. EP Patent App. EP20,000,921,568.

François Chollet et al. 2015. Keras. https://keras.io.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12(Aug):2493–2537.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks* 18(5-6):602–610.

Chinnappa Guggilla, Tristan Miller, and Iryna Gurevych. 2016. Cnn-and lstm-based claim classification in online user comments. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. pages 2740–2751.

Rie Johnson and Tong Zhang. 2014. Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058* .

Rie Johnson and Tong Zhang. 2015. Semi-supervised convolutional neural networks for text categorization via region embedding. In *Advances in neural information processing systems*. pages 919–927.

Piotr Kaminski and Kate Robu. 2016. A best practice model for bank compliance. http://www.mckinsey.com/business-functions/risk/our-insights/a-best-practice-model-for-bank-compliance.

Aurangzeb Khan, Baharum Baharudin, Lam Hong Lee, and Khairullah Khan. 2010. A review of machine learning algorithms for text-documents classification. *Journal of advances in information technology* 1(1):4–20.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* .

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *AAAI*. volume 333, pages 2267–2273.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101* .

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP 2014*. volume 14, pages 1532–43.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15(1):1929–1958. http://dl.acm.org/citation.cfm?id=2627435.2670313.

Financial & Risk Thomson Reuters. 2015. Risk management: Turn regulatory compliance into a business opportunity. https://financial.thomsonreuters.com/en/markets-industries/risk-management-tools.html?utm_campaign=e4&utm_medium=social&utm_source=FRblog&utm_content=DCraigPeakRegulatory.

Francesco Visin, Kyle Kastner, Kyunghyun Cho, Matteo Matteucci, Aaron Courville, and Yoshua Bengio. 2015. Renet: A recurrent neural network based alternative to convolutional networks. *arXiv preprint arXiv:1505.00393* .

Matthew D. Zeiler. 2012. Adadelta: An adaptive learning rate method. *CoRR* abs/1212.5701.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*. pages 649–657.