Inferring the source of official texts: can SVM beat ULMFiT?

Pedro Henrique Luz de Araujo[1], Teófilo Emidio de Campos[1], and Marcelo Magalhães Silva de Sousa[2]

[1] Departamento de Ciência da Computação (CiC), Universidade de Brasília (UnB)
[2] Tribunal de Contas do Distrito Federal, Zona Cívico-Administrativa
Brasília – DF, Brazil
pedrohluzaraujo@gmail.com, t.decampos@oxfordalumni.org, marcelomsousa@tc.df.gov.br

Abstract. Official Gazettes are a rich source of relevant information to the public. Their careful examination may lead to the detection of frauds and irregularities that may prevent mismanagement of public funds. This paper presents a dataset composed of documents from the Official Gazette of the Federal District, containing both samples with document source annotation and unlabeled ones. We train, evaluate and compare a transfer learning based model that uses ULMFiT with traditional bag-of-words models that use SVM and Naive Bayes as classifiers. We find the SVM to be competitive, its performance being marginally worse than the ULMFiT while having much faster train and inference time and being less computationally expensive. Finally, we conduct ablation analysis to assess the performance impact of the ULMFiT parts.

Keywords: Text classification · Language models · Transfer learning.

1 Introduction

Government Gazettes are a great source of information of public interest. These government maintained periodical publications disclose a myriad of matters, such as contracts, public notices, financial statements of public companies, public servant nominations, public tenderings, public procurements and others. Some of the publications deal with public expenditures and may be subject to frauds and other irregularities.

On the other hand, it is not easy to extract information from Official Gazettes. The data is not structured, but available as natural language texts. In addition, the language used is tipically from the public administration domain, which can further complicate information extraction and retrieval.

Natural Language Processing (NLP) and Machine Learning (ML) techniques are great tools for obtaining information from official texts. NLP has been used to automatically extract and classify relevant entities in court documents [5, 3]. Other works [10, 6, 15, 12] explore the use of automatic summarization to

mitigate the amount of information legal professional have to process. Text classification has been utilized for decision prediction [1, 11], area of legal practice attribution [24] and fine-grained legal-issue classification. Some effort has been applied to the processing of Brazilian legal documents [20, 25, 17].

In this paper, we aim to identify the public body of origin of documents fom the Official Gazette of the Federal District. This is a first step in the direction of structuring the information present in Official Gazettes in order to enable more advanced applications such as fraud detection. Even though it is possible to extract the public entity that produced the document by using rules and regular expressions, such approach is not very robust: changes in document and phrase structure and spelling mistakes can greatly reduce its effectiveness. A machine learning approach may be more robust to such data variation.

Due to the small number of samples in our dataset, we explore the use of transfer learning for NLP. We choose ULMFiT [8] as the method due to it being less resource-intensive than other state-of-the-art approaches such as BERT [4] and GPT-2 [19]. Our main contributions are:

1. Making available to the community a dataset with labeled and unlabeled Official Gazette documents.
2. Training, evaluating and comparing a ULMFiT model to traditional bag-of-word models.
3. Performing an ablation analysis to examine the impact of the ULMFiT steps when trained on our data.

2 The Dataset

The data consists of 2,652 texts extracted from the Official Gazette of the Federal District[3]. Handcrafted regex rules were used to extract some information from each sample, such as publication date, section number, public body that issued the document and title. 797 of the documents were manually examined, from which 724 were found to be free of labeling mistakes. These documents were produced by 25 different public entities. We filter the samples with entities who have less than three samples, since this would mean no representation for the public body in either the training, validation or test set. As a result, we end up with 717 labeled examples from 19 public entities.

We then split these samples and the 1,928 unverified or incorrectly labeled texts into two separate datasets. The first for classification of public entity that produced the document and the other for the unsupervised training of a language model.

The classification dataset is formed by 717 pairs of document and its respective public entity of origin. We randomly sample 8/15 of the texts for the training set, 2/15 for the validation set and the remainder for the test set, which results in 384, 96 and 237 documents in each set, respectively. Figure 1 shows the class distribution in each set. The data is imbalanced: *Segurança Pública*, the

[3] Available at `https://www.dodf.df.gov.br/`.

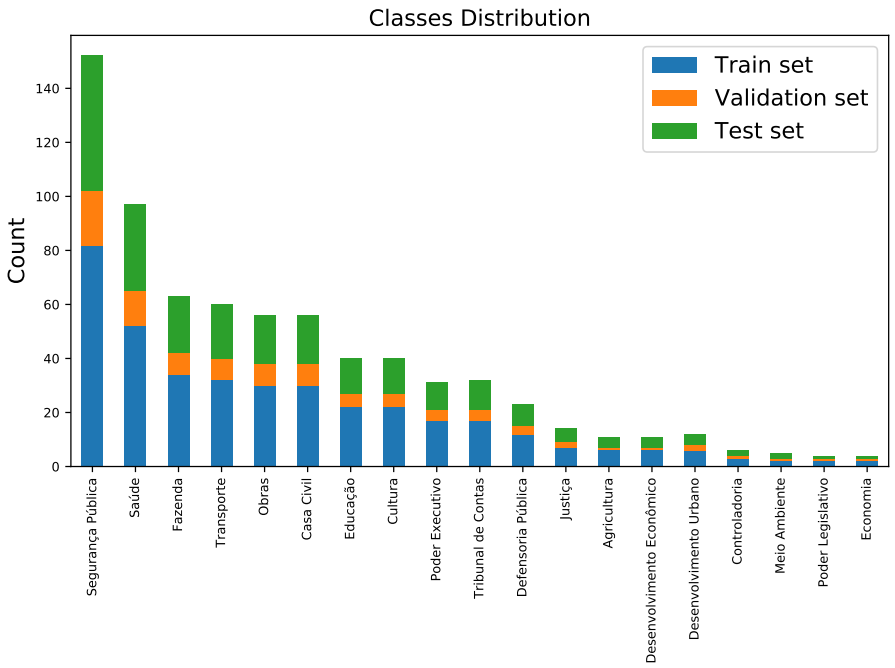

Fig. 1. Class counts for each dataset split.

most frequent class, contains more than 140 samples, while the least frequent classes are represented by less than 5 documents. We handle this by using F_1 score as the metric for evaluation and trying model-specific strategies to handle imbalance, as we discuss in Section 4.

Two of the 1,928 texts in the language model dataset were found to be empty and were dropped. From the remaining 1,926, 20% were randomly chosen for the validation set. The texts contain 984,580 tokens in total; after the split, there are 784,260 in the training set and 200,320 in the validation set. In this case we choose to not build a test set since we are not interested in an unbiased evaluation of the language model performance. The data is automatically labeled as an standard language model task where the label of each token is the following token in the sentence.

3 The Models

In this section we describe the transfer learning based approach to text classification used to classify the documents, the bag-of-words method used as a baseline and the preprocessing employed for both approaches.

3.1 Preprocessing

We first lowercase the text and use SentencePiece [14] to tokenize it. We chose SentencePiece because that was the tokenizer used for the pretrained language model (more about that on section 3.3), so using the same tokenization was fundamental to preserve vocabulary. We use the same tokenization for the baseline methods to establish a fair comparison of the approaches.

In addition, we add special tokens to the vocabulary to indicate unknown words, padding, beginning of text, first letter capitalization, all letters capitalization, character repetition and word repetition. Even though the text has been lowercased, these tokens preserve the capitalization information present in the original data. The final vocabulary is composed of 8,552 tokens, including words, subwords, special tokens and punctuation.

3.2 Baseline

For the baseline models, we experiment with two different bag-of-words text representation methods: tf-idf values and token counts. Both methods represent each document as a v-dimensional vector, where v is the vocabulary size. In the first case, the i-th entry of the vector is the tf-idf value of the i-th token in the vocabulary, while in the second case that value is simply the number of times the token appears in the document. Tf-idf values are computed according to the following equations:

$$\text{tf-idf}(t, d) = \text{tf}(t, d) \times \text{idf}(t) \tag{1}$$

$$\text{idf}(t) = \log \frac{1 + n}{1 + \text{count}(t)} + 1\,, \tag{2}$$

where $\text{tf}(t, d)$ is the frequency of term t in document d, n is the total of documents in the corpus, and $\text{count}(t)$ is the number of documents that contain term t. All document vectors are normalized to have unit Euclidean norm.

We use the obtained bag-of-words to train a shallow classifier. We experiment with both Support Vector Machines (SVM) [7] with linear kernel and Naive Bayes classifiers.

3.3 Transfer Learning

We use Universal Language Model Fine-Tuning (ULMFiT) [8] to leverage information contained in the unlabeled language model dataset. This method of inductive transfer learning was shown to require much fewer labeled examples to match the performance of training from scratch.

ULMFiT amounts to three stages:

Language model pre-training We use a bidirectional Portuguese language model[4] trained on a Wikipedia corpus composed of 166,580 articles, with a total of 100,255,322 tokens. The tokenization used was the same as ours. The

[4] Available at `https://github.com/piegu/language-models/tree/master/models`.

model architecture consists of a 400-dimensional embedding layer, followed by four Quasi-Recurrent Neural Network (QRNN [2]) layers with 1550 hidden parameters each and a final linear classifier on top. QRNN layers alternate parallel convolutional layers and a recurrent pooling function, outperforming LSTMs of same hidden size while being faster at training time and inference.

Language model fine-tuning We fine-tune the forward and backward pretrained general-domain Portuguese language models on our unlabeled dataset, since the latter comes from the same distribution as the classification task data, while the former does not. As in the ULMFiT paper, we use discriminative fine-tuning [8], where instead of using the same learning rate for all layers of the model, different learning rates are used for different layers. We employ cyclical learning rates [22] with cosine annealing to speed up training.

Classifier fine-tuning To train the document classifier, we add two linear blocks to the language models, each block composed of batch normalization [9], dropout [23] and a fully-connected layer. The first fully-connected layer has 50 units and ReLU [18] activation, while the second one has 19 units and is followed by a softmax activation that produces the probability distribution over the classes. The final prediction is the average of the forward and backwards models. The input to the linear blocks is the concatenation of the hidden state of the last time step $\boldsymbol{h}_T$ with the max-pooled and the average-pooled hidden states of as many time steps as can be fit in GPU memory $\boldsymbol{H} = \{\boldsymbol{h}_1, \cdots, \boldsymbol{h}_T\}$. That is, the input to the linear blocks $\boldsymbol{h}_c$ is:

$$\boldsymbol{h}_c = \text{concat}(\boldsymbol{h}_t, \text{maxpool}(\boldsymbol{H}), \text{averagepool}(\boldsymbol{H}))\,. \tag{3}$$

4 Experiments

In this section we describe the training procedure and hyperparameters used. All experiments were executed on a Google Cloud Platform n1-highmem-4 virtual machine with a Nvidia Tesla P4 GPU.

4.1 Baseline

To find the best set of hyperparameter values we use random search and evaluate the model on the validation set. Since we experiment with two classifiers (SVM and Naive Bayes) and two text vectorizers (tf-idf values and token counts), we have four model combinations: tf-idf and Naive Bayes, tf-idf and SVM, token counts and Naive Bayes; and token counts and SVM. For each of these 4 scenarios we train 100 models, each iteration with random hyperparameter values.

Vectorizers For both the tf-idf and token counts vectorizers we tune the same set of hyperparameters: n-gram range (only unigrams, unigrams and bigrams, unigrams to trigrams), maximum document frequency token cutoff (50%, 80% and 100%), minimum number of documents for token cutoff (1, 2 and 3 documents).

Naive Bayes We tune the smoothing prior α on a exponential scale from 10^{-4} to 1. We also choose between fitting the prior probabilities, which could help with the class imbalance, and just using an uniform prior distribution.

SVM In the SVM case, we tune two hyperparameters. We sample the regularization parameter C from an exponential scale from 10^{-3} to 10. In addition, we choose between applying weights inversely proportional to class frequencies to compensate class imbalance and giving all classes the same weight.

4.2 Transfer Learning

To tune the best learning rate in both the language model fine-tuning and classifier training scenarios, we use the learning rate range test [21], where we run the model through batches while increasing the learning rate value, choosing the learning rate value that corresponds to the steepest decrease in validation loss. We use Adam [13] as the optimizer.

We fine-tune the top layer of the forward and backwards language models for one one cycle of 2 epochs and then train all layers for one cycle of 10 epochs. We use a batch size of 32 documents, weight decay [16] of 0.1, backpropagation through time of length 70 and dropout probabilities of 0.1, 0.6, 0.5 and 0.2 applied to embeddings inputs, embedding outputs, QRNN hidden-to-hidden weight matrix and QRNN output, respectively, following previous work [8].

In the case of the backward and forward classifiers, in order to prevent catastrophic forgetting by fine-tuning all layers at once, we gradually unfreeze [8] the layers starting from the last layer. Each time we unfreeze a layer we fine-tune for one cycle of 10 epochs. We use a batch size of 8 documents, weight decay of 0.3, backpropagation through time of length 70 and the same dropout probabilities used for the language model fine-tuning scaled by a factor of 0.5.

Similarly to the SVM experiments, in order to handle data imbalance we try applying weights inversely proportional to class frequencies. Nevertheless, this did not contribute to significant changes in classification metrics.

5 Results

Table 1 reports, for each model trained, test set F_1 scores for each class. Due to the small size of the classification dataset, some class-specific scores are noisy because of their rarity, so we also present the average and weighted by class frequency F_1 values and the model accuracy. For the baseline models, we present

Table 1. Classification results (in %) on the test set.

Class	NB	SVM	F-ULMFiT	B-ULMFiT	F+B-ULMFiT	Count
Casa Civil	72.22	74.29	82.35	83.33	**85.71**	18
Controladoria	**80.00**	**80.00**	**80.00**	0.00	66.67	2
Defensoria Pública	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	8
Poder Executivo	80.00	81.82	78.26	**90.91**	86.96	10
Poder Legislativo	40.00	**100.00**	**100.00**	**100.00**	**100.00**	1
Agricultura	28.57	75.00	**85.71**	75.00	**85.71**	4
Cultura	**91.67**	**91.67**	88.00	**91.67**	**91.67**	13
Desenv. Econômico	**66.67**	**66.67**	28.57	33.33	33.33	4
Desenv. Urbano	75.00	**85.71**	75.00	66.67	**85.71**	4
Economia	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	1
Educação	70.00	**75.00**	72.00	66.67	**75.00**	13
Fazenda	85.71	88.37	86.36	**90.48**	**90.48**	21
Justiça	**80.00**	75.00	66.67	75.00	66.67	5
Obras	84.85	85.71	87.50	87.50	**90.91**	18
Saúde	91.43	93.94	**95.38**	91.43	94.12	32
Segurança Pública	**97.03**	95.24	95.24	96.15	94.34	50
Transporte	91.89	95.00	87.18	**95.24**	**95.24**	20
Meio Ambiente	80.00	**100.00**	66.67	66.67	66.67	2
Tribunal de Contas	**100.00**	**100.00**	95.65	**100.00**	**100.00**	11
Average F1	79.74	**87.55**	82.66	79.48	84.69	237
Weighted F1	86.86	89.17	87.46	88.14	**89.74**	237
Accuracy	86.92	89.45	87.76	89.03	**90.30**	237

results using the tf-idf text vectorizer, which performed better than the count vectorizer on the validation set. F-ULMFiT, B-ULMFiT and F+B-ULMFiT indicate the forward ULMFiT model, the backward counterpart and their ensemble, respectively.

All models performed better than a classifier that simply chooses the most common class, which would yield average and weighted F_1 scores of 7.35 and 1.83 and an accuracy of 21.10. The SVM and ULMFiT models outperformed the Naive Bayes classifier across almost all categories. All models seem to achieve good results, with weighted F_1 scores and accuracies approaching 90.00%, though we do not have a human performance benchmark for comparison.

Despite the SVM's average F_1 score being higher than the ULMFiT's, the latter has greater weighted F_1 score and accuracy, with a corresponding reduction of 8.06% on test error rate. That being said, the SVM has some advantages. First, it is much faster to train. While the SVM took less than two seconds to train, the ULMFiT model took more than half an hour. In addition, the ULMFiT approach greatly depends on GPU availability, otherwise training would take much more time.

Furthermore, SVM training is very straightforward, while the transfer learning scenario requires three different steps with many parts that need tweaking

Table 2. Ablation scenarios results (in %) on the test set.

Model	Average F1	Weighted F1	Accuracy
No gradual unfreeezing (f)	82.34 (-0.32)	89.46 (+2.00)	89.87 (+2.11)
No gradual unfreeezing (b)	80.8 (+1.32)	89.07 (+9.03)	89.87 (+0.84)
No gradual unfreeezing (f+b)	82.76 (-1.93)	89.66 (- 0.08)	89.87 (-0.43)
Last layer fine-tuning (f)	63.30 (-19.36)	77.39 (-10.07)	78.90 (-8.86)
Last layer fine-tuning (b)	60.48 (-19.00)	77.03 (-11.11)	78.48 (-10.55)
Last layer fine-tuning (f+b)	66.37 (-18.32)	79.60 (-10.14)	81.01 (-9.29)
No LM fine-tuning (f)	28.05 (-54.61)	47.24 (-40.22)	53.59 (-34.17)
No LM fine-tuning (b)	27.32 (-52.16)	39.24 (-48.90)	50.63 (-38.40)
No LM fine-tuning (f+b)	31.48 (-53.21)	46.06 (-43.68)	55.27 (-35.03)
Direct transfer (f)	11.78 (-70.88)	24.33 (-63.13)	32.07 (-55.69)
Direct transfer (b)	8.33 (-71.15)	14.01 (-74.13)	27.85 (-61.18)
Direct transfer (f+b)	11.54 (-73.15)	24.00 (-65.74)	34.60 (-55.70)

(gradual unfreezing, learning rate schedule, discriminative fine-tuning). Consequently, not only the ULMFiT model has more hyperparameters to be tuned, each parameter search iteration is computationally expensive—the time it takes to train one ULMFiT model is enough to train more than 1,000 SVM models with different configurations of hyper-parameters.

5.1 Ablation Analysis

In this section we analyze the individual impact of ULMFiT's parts on our data. We do so by running experiments on four different scenarios. We use the same hyperparameters as in the complete ULMFiT case and train for the same number of iterations in order to establish a fair comparison. Table 2 presents the results and the difference between the scenario result and the original performance, taking into consideration if it is the forward, backward or ensemble case.

No gradual unfreezing This scenario's training procedure is almost identical to the previously presented, with the exception that gradual unfreezing is not used. In the classifier fine-tuning step though, we instead fine-tune all layers at the same time. This was the least contributing to the performance, with minor reductions to our metrics in the ensemble case.

Last layer fine-tuning This scenario is similar to the previous one in the sense that we do not perform gradual unfreezing. But while there we fine-tuned all layers, here we treat the network as a feature extractor and fine-tune only the classifier. We see a sharp decrease in performance across all metrics, suggesting that the QRNN network, even though the language model was fine-tuned on domain data, does not perform well as a feature extractor for document classification. That is, to train a good model it is imperative to fine-tune all layers.

No language model fine-tuning Here we skip the language model fine-tuning step and instead train the classifier directly from the pre-trained language model, using gradual unfreezing just like in the original model. This results in a great decline in performance, with decreases ranging from about 30 to more than 50 percentual points. Therefore, for our data, training a language model on general domain data is not enough; language model fine-tuning on domain data is essential. This may be due to differences in vocabulary and word distribution between general and official text domains.

Direct transfer In this scenario we go one step further than in the previous one: we start from the pre-trained language model and do not fine-tune it. They differ because in the classifier fine-tuning step we do not perform gradual unfreezing, but train all layers at the same time. This results in a even greater performance decrease. The lack of gradual unfreezing here is much more dramatic than in the first scenario. We hypothesize that the language model fine-tuning may mitigate the effects or decrease the possibility of catastrophic forgetting.

Averaging forward and backward predictions In almost all cases, averaging the forward and backward models predictions results in more accurate results than either of the single models. One possible way of further experimenting is trying other methods of combining the directional outputs.

6 Conclusion

This paper examines the use of ULMFiT, a inductive transfer learning method for natural language applications, to identify the public entity that originated Official Gazette texts. We compare the performance of ULMFiT with simple bag-of-words baselines and perform an ablation analysis to identify the impact of gradual unfreezing, language model fine-tuning and the use of the fine-tuned language model as a text feature extractor.

Despite being a state-of-the-art technique, the use of ULMFiT correspond to a small increase in classification accuracy when compared to the SVM model. Considering the faster training time, simpler training procedure and easier parameter tuning of SVM, this traditional text classification method is still competitive with modern deep learning models.

Finally, our ablation analysis shows that the combination of language model fine-tuning and gradual unfreezing is extremely beneficial. It also suggests that language models, even after fine-tuned on domain data, are not good feature extractors and should be trained also on classification data.

Acknowledgements This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. TdC received support from Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), grant PQ 314154/2018-3. We are also grateful for the support from Fundação de Apoio à Pesquisa do Distrito Federal (FAPDF).

References

1. Aletras, N., Tsarapatsanis, D., Preotiuc-Pietro, D., Lampos, V.: Predicting judicial decisions of the european court of human rights: A natural language processing perspective. PeerJ in Computer Science **2016** (10 2016). https://doi.org/10.7717/cs.93
2. Bradbury, J., Merity, S., Xiong, C., Socher, R.: Quasi-recurrent neural networks. CoRR **abs/1611.01576** (2016), http://arxiv.org/abs/1611.01576
3. Cardellino, C., Teruel, M., Alonso Alemany, L., Villata, S.: A low-cost, high-coverage legal named entity recognizer, classifier and linker. In: Proceedints of the 16th International Conference on Artificial Intelligence and Law (ICAIL). London, United Kingdom (June 2017), preprint available from https://hal.archives-ouvertes.fr/hal-01541446
4. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR **abs/1810.04805** (2018), http://arxiv.org/abs/1810.04805
5. Dozier, C., Kondadadi, R., Light, M., Vachher, A., Veeramachaneni, S., Wudali, R.: Named entity recognition and resolution in legal text. In: Semantic Processing of Legal Texts, pp. 27–43. Springer (2010)
6. Galgani, F., Compton, P., Hoffmann, A.: Combining different summarization techniques for legal text. In: Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data. pp. 115–123. HYBRID, Association for Computational Linguistics (ACL), Stroudsburg, PA, USA (2012), http://dl.acm.org/citation.cfm?id=2388632.2388647
7. Hearst, M.A.: Support vector machines. IEEE Intelligent Systems **13**(4), 18–28 (Jul 1998)
8. Howard, J., Ruder, S.: Fine-tuned language models for text classification. CoRR **abs/1801.06146** (2018), http://arxiv.org/abs/1801.06146
9. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proceedings of the 32nd International Conference on Machine Learning - Volume 37. pp. 448–456. JMLR.org (2015), http://dl.acm.org/citation.cfm?id=3045118.3045167
10. Kanapala, A., Pal, S., Pamula, R.: Text summarization from legal documents: a survey. Artificial Intelligence Review (Jun 2017). https://doi.org/10.1007/s10462-017-9566-2
11. Katz, D.M., Bommarito, Michael J, I., Blackman, J.: A general approach for predicting the behavior of the Supreme Court of the United States. PLoS ONE (4 2017). https://doi.org/10.1371/journal.pone.0174698
12. Kim, M.Y., Xu, Y., Goebel, R.: Summarization of legal texts with high cohesion and automatic compression rate. In: New frontiers in artificial intelligence. Springer (2013)
13. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015)
14. Kudo, T., Richardson, J.: SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP). pp. 66–71. Association for Computational Linguistics (ACL), Brussels, Belgium (Nov 2018)
15. Kumar, R., Raghuveer, K.: Legal document summarization using latent Dirichlet allocation. International Journal of Computer Science and Telecommunications **3**, 114–117 (2012)

16. Loshchilov, I., Hutter, F.: Fixing weight decay regularization in Adam. CoRR **abs/1711.05101** (2017), http://arxiv.org/abs/1711.05101
17. Luz de Araujo, P.H., de Campos, T.E., de Oliveira, R.R.R., Stauffer, M., Couto, S., Bermejo, P.: LeNER-Br: a dataset for named entity recognition in Brazilian legal text. In: International Conference on the Computational Processing of Portuguese (PROPOR). pp. 313–323. Lecture Notes on Computer Science (LNCS), Springer, Canela, RS, Brazil (September 24-26 2018), https://cic.unb.br/~teodecampos/LeNER-Br/
18. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICLR). pp. 807–814. Omnipress, USA (2010), https://icml.cc/Conferences/2010/papers/432.pdf
19. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI blog **1**(8) (February 2019), https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf
20. da Silva, N.C., Braz, F.A., de Campos, T.E., Gusmao, D., Chaves, F., Mendes, D., Bezerra, D., Ziegler, G., Horinouchi, L., Ferreira, M., Carvalho, G., Fernandes, R.V.C., Peixoto, F.H., Filho, M.S.M., Sukiennik, B.P., Rosa, L.S., Silva, R.Z.M., Junquilho, T.A.: Document type classification for Brazil's supreme court using a convolutional neural network. In: 10th International Conference on Forensic Computer Science and Cyber Law (ICoFCS). Sao Paulo, Brazil (October 29-30 2018). https://doi.org/10.5769/C2018001, winner of the best paper award.
21. Smith, L.N.: No more pesky learning rate guessing games. CoRR **abs/1506.01186** (2015), http://arxiv.org/abs/1506.01186
22. Smith, L.N., Topin, N.: Super-convergence: Very fast training of residual networks using large learning rates. CoRR **abs/1708.07120** (2017), http://arxiv.org/abs/1708.07120
23. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (Jan 2014), http://dl.acm.org/citation.cfm?id=2627435.2670313
24. Şulea, O.M., Zampieri, M., Vela, M., van Genabith, J.: Predicting the law area and decisions of french supreme court cases. In: Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP). pp. 716–722. INCOMA Ltd. (2017)
25. de Vargas Feijó, D., Moreira, V.P.: Rulingbr: A summarization dataset for legal texts. In: Villavicencio, A., Moreira, V., Abad, A., Caseli, H., Gamallo, P., Ramisch, C., Gonçalo Oliveira, H., Paetzold, G.H. (eds.) Computational Processing of the Portuguese Language. pp. 255–264. Springer International Publishing, Cham (2018)