

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

Pedro Henrique Luz de Araujo<sup>1</sup>   Teófilo Emidio de Campos<sup>1</sup>  
Marcelo Magalhães Silva de Sousa<sup>2</sup>

<sup>1</sup>Department of Computer Science, University of Brasília, Brasília – DF, Brazil

<sup>2</sup>Tribunal de Contas do Distrito Federal, Zona Cívico-Administrativa, Brasília – DF, Brazil

*pedrohluzaraujo@gmail.com, t.decampos@st-annes.oxon.org,  
marcelomsousa@tc.df.gov.br*

February 14, 2020

# Overview

- 1 Introduction
- 2 The dataset
- 3 The models
- 4 Experiments
- 5 Results
- 6 Conclusion

# Introduction

# Motivation

- Government Gazettes are a great source of information of public interest:
  - ▶ Nominations, contracts, public notices.
  - ▶ Public expenditures may be subject to frauds and irregularities.
- Difficulties:
  - ▶ Unstructured data.
  - ▶ Domain-specific language (official texts).

# Examples

## Excerpt 1

O GOVERNADOR DO DISTRITO FEDERAL, no uso das atribuições que lhe confere o artigo 100, incisos XXVI e XXVII, da Lei Orgânica do Distrito Federal, resolve [...]

## Excerpt 2

Presidente da COVED, acolhendo os pareceres inseridos nos processos abaixo, declara habilitados para a venda à PRAZO os itens a seguir: [...]

## Excerpt 2

[...] TORNAR PÚBLICO o resultado das investigações constantes nos processos dos servidores listados abaixo e que se configuraram em acidente de serviço, sem dano, nos termos do artigo 23, § 1º, inciso IV, do Decreto nº 34.023, de 10 de dezembro de 2012, observando-se a seguinte ordem: número do processo, nome e matrícula. [...]

# Objectives

- Identify the public body of origin of documents from the Official Gazette of the Federal District through Machine Learning.
  - ▶ First step in the direction of structuring the information present in Official Gazettes.
  - ▶ Extracting the public entity that produced the document by using rules and regular expressions is not robust.
- Leverage unlabelled samples through transfer learning (ULMFiT [Howard and Ruder, 2018]) to compensate for the small number of annotated instances.

# Contributions

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

# The dataset



# The dataset I

- 2,652 texts extracted from the Official Gazette of the Federal District.<sup>1</sup>
- Handcrafted regex rules to extract some information from each sample: publication date, section number, public body that issued the document, title and others.
- 797 texts manually examined: 724 free of labelling mistakes.

# The dataset II

- Documents originated from 25 different public entities.
- Filter out entities with less than three samples.
  - ▶ Final count: 717 labelled examples from 19 different public entities.
- Divide the data into two separate parts:
  - ▶ 717 labelled examples for public entity of origin classification.
  - ▶ 1,928 unverified or incorrectly labelled samples for unsupervised training of a language model.

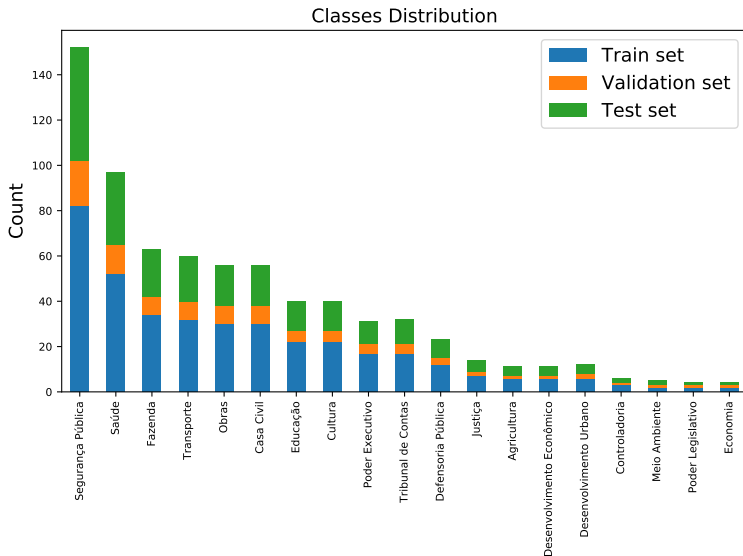
---

<sup>1</sup>Available at <https://www.dodf.df.gov.br/>.

# Classification data

- Randomly sample 8/15 (384) of the texts for the training set, 2/15 (96) for the validation set and 5/15 (237) for the test set.
- Imbalanced data: most frequent class (*Segurança Pública*) with 140 samples and least frequent classes with less than 5 documents.

# Data Distribution



# Language model data

- Drop two empty texts, totalizing 1,926 documents.
- 80 % for training and 20 % for validation.
  - ▶ No test set: no need for unbiased evaluation of the language model.
- Total of 984,580 tokens: 784,260 in the training set and 200,320 in the validation set.
- Standard language modelling task: label of each token is the following token in the sentence.

## The models

# Preprocessing

- Lowercase text and use SentencePiece [Kudo and Richardson, 2018] to tokenize.
  - ▶ The same used for the pretrained language model.
- Add special tokens for padding, first letter capitalization, all letters capitalization, character and word repetition etc.<sup>2</sup>
- Final vocabulary of 8,552 tokens, including words, subwords, special tokens and punctuation.

---

<sup>2</sup>List of special tokens used available at

<https://docs.fast.ai/text.transform.html>

- Two kinds of BOW:
  - ▶ tf-idf values;
  - ▶ token counts.
- Two classifiers:
  - ▶ Naïve Bayes (NB);
  - ▶ Support Vector Machines (SVM) with linear kernel.

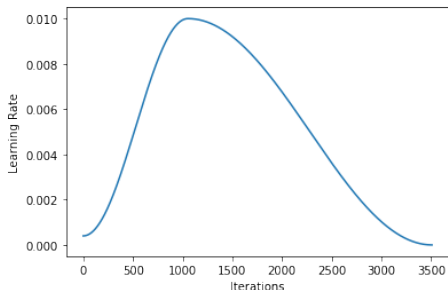


# Transfer learning I

- Language Model Fine-Tuning (ULMFiT) [Howard and Ruder, 2018] to leverage information contained in the unlabeled language model dataset. Three stages:
- **Language model pre-training**: we use a bidirectional Portuguese language model<sup>3</sup> trained on 166,580 Wikipedia articles (100,255,322 tokens).
  - ▶ Architecture: 400-dimensional embedding layer + 4 Quasi-Recurrent Neural Network (QRNN [Bradbury et al., 2016]) layers + linear classifier.

# Transfer learning II

- **Language model fine-tuning:** we fine-tune the forward and backward pre-trained language models on our unlabelled dataset.
  - ▶ We use discriminative fine-tuning (different learning rates for different layers) and cyclical learning rates [Smith and Topin, 2017] with cosine annealing to speed up training.




# Transfer learning III

- **Classifier fine-tuning:** we add two linear blocks to the language models (batch normalization [Ioffe and Szegedy, 2015] + dropout [Srivastava et al., 2014] + FC layer).
  - ▶ Final prediction is the average of the forward and backward models.
  - ▶ Let  $\mathbf{h}_t$  be the hidden state of the last time step, and  $\mathbf{H} = \{\mathbf{h}_{t-T}, \dots, \mathbf{h}_t\}$ , be the hidden states of as many time steps as can be fit in GPU memory. Then, the input to the linear blocks  $\mathbf{h}_c$  is:

$$\mathbf{h}_c = \text{concat}(\mathbf{h}_t, \text{maxpool}(\mathbf{H}), \text{averagepool}(\mathbf{H})). \quad (1)$$

---

<sup>3</sup>Available at

<https://github.com/piegu/language-models/tree/master/models> 

# Experiments

# Baseline I

- Random search + evaluation on validation set to find best hyperparameter values.
- Four scenarios: tf-idf + NB; tf-idf + SVM; counts + NB; counts + SVM.
- For each scenario we train 100 models with different randomly assigned hyperparameter values.

- Vectorizers hyperparameters:
  - ▶ n-gram range (only unigrams, unigrams and bigrams, unigrams to trigrams).
  - ▶ maximum document frequency (50%, 80% and 100%)
  - ▶ minimum number of documents (1, 2 and 3 documents)
- NB hyperparameters:
  - ▶ Smoothing prior  $\alpha$  uniformly sampled from  $\{0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1\}$ .
  - ▶ Fitting prior probability vs. using uniform prior distribution.
- SVM hyperparameters:
  - ▶ Regularization parameter  $C$  sampled from  $\{0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, 3, 10\}$
  - ▶ Applying weights inversely proportional to class frequencies vs. unitary weight for all classes.

# Transfer learning I

- We use the learning rate range test [Smith, 2015] to tune the learning rate.
- Adam is used as the optimizer.
- Language model fine-tuning:
  - ▶ We fine-tune the top layer of the forward and backward language models for one cycle of two epochs and then train all layers for one cycle of ten epochs.
  - ▶ Batch size of 32 documents, weight decay of 0.1, BPTT 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.

- Classifier fine-tuning:
  - ▶ We employ gradual unfreezing to prevent catastrophic forgetting:
    - ★ We unfreeze one layer at a time, starting from the last, each time fine-tuning for one cycle of 10 epochs.
  - ▶ Batch size of 8 documents, weight decay of 0.3, BPTT of length 70 and the same dropout probabilities used for the language model fine-tuning scaled by a factor of 0.5.



# Results

Table: Classes  $F_1$  scores (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	<b>85.71</b>	18
Controladoria	<b>80.00</b>	<b>80.00</b>	<b>80.00</b>	0.00	66.67	2
Defensoria Pública	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	8
Poder Executivo	80.00	81.82	78.26	<b>90.91</b>	86.96	10
Poder Legislativo	40.00	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	1
Agricultura	28.57	75.00	<b>85.71</b>	75.00	<b>85.71</b>	4
Cultura	<b>91.67</b>	<b>91.67</b>	88.00	<b>91.67</b>	<b>91.67</b>	13
Desenv. Econômico	<b>66.67</b>	<b>66.67</b>	28.57	33.33	33.33	4
Desenv. Urbano	75.00	<b>85.71</b>	75.00	66.67	<b>85.71</b>	4
Economia	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	<b>100.00</b>	1
Educação	70.00	<b>75.00</b>	72.00	66.67	<b>75.00</b>	13
Fazenda	85.71	88.37	86.36	<b>90.48</b>	<b>90.48</b>	21
Justiça	<b>80.00</b>	75.00	66.67	75.00	66.67	5
Obras	84.85	85.71	87.50	87.50	<b>90.91</b>	18
Saúde	91.43	93.94	<b>95.38</b>	91.43	94.12	32
Segurança Pública	<b>97.03</b>	95.24	95.24	96.15	94.34	50
Transporte	91.89	95.00	87.18	<b>95.24</b>	<b>95.24</b>	20
Meio Ambiente	80.00	<b>100.00</b>	66.67	66.67	66.67	2
Tribunal de Contas	<b>100.00</b>	<b>100.00</b>	95.65	<b>100.00</b>	<b>100.00</b>	11
Average F	79.74	<b>87.55</b>	82.66	79.48	84.69	237
Weighted F1	86.86	89.17	87.46	88.14	<b>89.74</b>	237
Accuracy	86.92	89.45	87.76	89.03	<b>90.30</b>	237

## Results II

- All models performed better than a majority class classifier, which yields  $F_1$  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.
- $F_1$  scores and accuracies approaching 90.00% indicate good results, though we do not have a human performance benchmark for comparison.

## Results III

- SVM and ULMFiT scores are comparable: the former has greater average  $F_1$  score while the latter wins at weighted  $F_1$  score and accuracy.
- SVM has some advantages:
  - ▶ Time: SVM takes less than two seconds to train on CPU, while ULMFiT takes more than half an hour on GPU.
  - ▶ Simplicity: SVM training is straightforward, while ULMFiT requires three different steps with many parts that need tweaking (gradual unfreezing, learning rate schedule, discriminative fine-tuning).
- Consequence: ULMFiT has more hyperparameters to tune and each search iterations is 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.

# Ablation analysis I

- Four scenarios using the same hyperparameters and trained for the same number of iterations:
  - ▶ **No gradual unfreezing**: we fine-tune all layers at the same time.
  - ▶ **Last layer fine-tuning**: we treat language model as feature extractor and fine-tune only the linear blocks on top (classifier).
  - ▶ **No language model fine-tuning**: we skip the language model fine-tuning and train the classifier directly from the pretrained language model, using gradual unfreezing.
  - ▶ **Direct transfer**: similar to the previous step, though we train all layers at the same time instead of employing gradual unfreezing.

# Ablation analysis II

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

Model	Average F1	Weighted F1	Accuracy
No gradual unfreezing (f)	82.34 (-0.32)	89.46 (+2.00)	89.87 (+2.11)
No gradual unfreezing (b)	80.8 (+1.32)	89.07 (+9.03)	89.87 (+0.84)
No gradual unfreezing (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)

- **Averaging forward and backward predictions:** Averaging the forward and backward models predictions results usually improves results.
  - ▶ May be worth to try other methods of combining the directional outputs.




## Conclusion



# Conclusion

- We have compared the performance of ULMFiT and BOW methods when identifying the public entity that originated Official Gazette texts.
- We have performed an 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.
- We have found that SVM is competitive with ULMFiT, a SOTA technique, when considering its simpler training procedure, parameter tuning and faster training time.
- The ablation analysis indicate the combination of LM tuning and gradual unfreezing is beneficial. On the other hand, LMs, even fine-tuned on domain data, are not good feature extractors.

# Referências I

-  Bradbury, J., Merity, S., Xiong, C., and Socher, R. (2016).  
Quasi-recurrent neural networks.  
*CoRR*, abs/1611.01576.
-  Howard, J. and Ruder, S. (2018).  
Fine-tuned language models for text classification.  
*CoRR*, abs/1801.06146.
-  Ioffe, S. and Szegedy, C. (2015).  
Batch normalization: Accelerating deep network training by reducing internal covariate shift.  
*In Proceedings of the 32nd International Conference on Machine Learning - Volume 37*, pages 448–456. JMLR.org.

## Referências II



Kudo, T. and Richardson, J. (2018).

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)*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics (ACL).



Smith, L. N. (2015).

No more pesky learning rate guessing games.

*CoRR*, abs/1506.01186.



Smith, L. N. and Topin, N. (2017).

Super-convergence: Very fast training of residual networks using large learning rates.

*CoRR*, abs/1708.07120.

## Referências III



Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014).

Dropout: A simple way to prevent neural networks from overfitting.  
*J. Mach. Learn. Res.*, 15(1):1929–1958.