# NLP applied to judicial decisions parsing by Predilex

AL Houceine KILANI, Marwan WEHAIBA EL KHAZEN
Ecole Normale supérieure Paris-Saclay
houcinekila@gmail.com, mwek1991@gmail.com

March 24, 2020

## Abstract

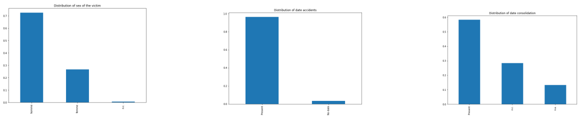
In this report, we present the methods we applied to classify French juridical documents. This challenge was proposed by Predilex in the framework of Pr. Mallat's course on multi-scale models and DNNs. We achieved a final score of 77.52%.

## 1 Introduction

Predilex has given us a set of legal texts recounting the major events and decisions made in court cases (a person filing an insurance claim due to an accident, a conflict between two individuals, or between an individual and a company, etc.) We are asked to extract the `accident date`, `consolidation date` and `sex of the victim`[1], if present in the text. If there is no accident date in the corpus, 'n.c.' should be returned. If there is no consolidation date in the text, we should distinguish between two cases : either the victim is deceased, in which case, 'n.a.' must be returned, else 'n.c.' should be the result.

### 1.1 Classes distribution

We first acquaint ourselves with the data. There is an imbalance of the different targets. For example, over 70% of the cases are assigned to the male sex (see figure 1).



(a) Sex of victim (b) accident date (c) consolidation date

Figure 1: Distribution of labels to predict

### 1.2 First intuitions

As a first step, below are the intuitions constituting our prior information.

#### 1.2.1 Looking for the rows of interest

We visually explore the rows of some of the texts, in order to spot where the "story-telling" begins. Since there is no consistency over all the texts, we used an approximation (that works for 764 among the 770 texts in the training set) that consists in discarding the rows that fall before the first

[1] The notion of sex is entirely interchangeable with the notion of binary gender in the context of these legal texts

mention of the clerk (greffier/greffière in French)[2]

### 1.2.2 Dates order

In most cases if not all, the consolidation date should come after the accident date. We use this constraint to infer the dates from the texts. But we have to rely on a date to infer the other one: our choice was to rely on the accident date prediction to infer the consolidation date (the reason for this is simply that there are more instances of accident dates in the texts than consolidation dates, so we assumed the predictor will have better chances if we start with the accident class)

### 1.2.3 Terms discarding the dates

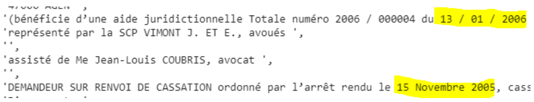
We build a list of terms that we call `terms_discarding_the_date` that we use for some predictors (used in the classical approach predictors) to filter out some dates in the inference phase and help improving prediction accuracy (this list contains terms such as `juge, audience, loi, court, naissance...`). But in the end, with the NN BERT approach section 4 we do not need it, as the classifier has higher performance and naturally discards those dates.

## 2 Text analysis, Pre-processing & Data set building

In this section we present our attempts at following guidelines from [3].

We first lowered the characters and removed accents from the text (we lost here the possibility of exploiting proper nouns to identify the sex of the victim - by using a 'dictionary of names'-based approach for example).

An important thing to notice is that the dates in the text are not all in the same format. So we standardized the dates in the same uniform format dd month yyyy (month all in letters).

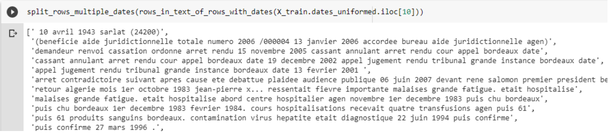


```
'(bénéficie d’une aide juridictionnelle Totale numéro 2006 / 000004 du 13 / 01 / 2006
'représenté par la SCP VIMONT J. ET E., avoués ',
'',
'assisté de Me Jean-Louis COUBRIS, avocat ',
'',
'DEMANDEUR SUR RENVOI DE CASSATION ordonné par l’arrêt rendu le 15 Novembre 2005, cass
```

Figure 2: Different formats for dates

We included the possibility to remove the stop-words and use stemming too. The stemming made the pre-processing of the data too slow so we did not use it at all in our approaches. As for the stop-words, we saw that removing them gave good results with the **Classical approaches** section 3, but keeping them worked better for BERT based approaches.

The sentences in any given text obviously have different lengths. We scanned the rows, keeping the rows that contain dates only (easier after the dates are standardised in the same format) and then kept $T$ tokens only before and after the date found. If a row contained more than one date, we assumed it might share some of the same tokens in its context with other dates. We then built a training set constituted of contexts of a maximum of 25 tokens (whenever there were less, we used padding) to label the corresponding row.



```
split_rows_multiple_dates(rows_in_text_of_rows_with_dates(X_train.dates_uniformed.iloc[10]))
```

```
[' 10 avril 1943 sarlat (24200)',
 '(beneficie aide juridictionnelle totale numero 2006 /000004 13 janvier 2006 accordee bureau aide juridictionnelle agen)',
 'demandeur renvoi cassation ordonne arret rendu 15 novembre 2005 cassant annulant arret rendu cour appel bordeaux date',
 'cassant annulant arret rendu cour appel bordeaux date 19 decembre 2002 appel jugement rendu tribunal grande instance bordeaux date',
 'appel jugement rendu tribunal grande instance bordeaux date 13 fevrier 2001 ',
 'arret contradictoire suivant apres cause ete debattue plaidee audience publique 06 juin 2007 devant rene salomon premier president be
 'retour algerie mois 1er octobre 1983 jean-pierre x... ressentait fievre importante malaises grande fatigue. etait hospitalise',
 'malaises grande fatigue. etait hospitalise abord centre hospitalier agen novembre 1er decembre 1983 puis chu bordeaux',
 'puis chu bordeaux 1er decembre 1983 fevrier 1984. cours hospitalisations recevait quatre transfusions agen puis 61',
 'puis 61 produits sanguins bordeaux. contamination virus hepatite etait diagnostique 22 juin 1994 puis confirme',
 'puis confirme 27 mars 1996 .',
```

Figure 3: Rows extracted from a single text

[2] This trick is not needed in the BERT approach as it is done naturally, but we do it for all methods anyway.

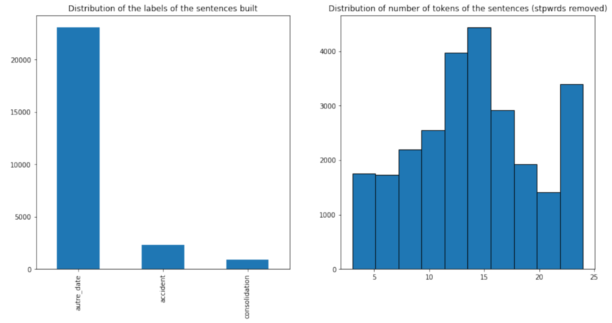


Figure 4: Left : Distribution of labels of the training set build (all the rows from all the texts) | Right : distribution of the number of tokens of each row | T=10

# 3 Classical approaches

'Classical', in our framework, stands for methods that do not use neural networks. We tested the major approaches : a Tf-idf approach[6], a dictionary approach and a Word Vectors approach. The latter falls in between the classical and NN solutions since the vectors were built using neural networks -Cf Word2Vec-[3]. But this last solution was computationally expensive and did not give satisfying good results, so we decided not to carry on using and analysing it.

## 3.1 Tf-idf based approaches

### 3.1.1 Tf-idf

Term frequency–inverse document frequency is a numerical statistic that is intended to reflect how important a word is to a document in a collection. The Tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general. Naturally, the tf-idf can discard the stop words because they are frequent among all the texts, but we force their filtering anyway, to lower the dimensionality of the matrix obtained.

### 3.1.2 Our approach for the dates predictions

We compute the mean of the tf-idf vectors constituting the context and fit a classifier of those build vectors. We tested multiple configurations : logistic regression versus SVM classifier, two classes (predict if the context is an accident context or a consolidation) versus three classes (where we build a third class consisting of 'other' dates such as dates of laws, court hearings, birth dates...). (see the **Summary section 5**).

Based on the prediction probability, we manually set thresholds to determine whether or not to choose the prediction as being in an accident context or in a consolidation context (using the probability of the third class).

### 3.1.3 Our approach for the sex prediction

We initially used a naive approach similar to the one used in the baseline, where we count occurrences of feminine versus masculine words.

## 3.2 Dictionary-based approaches

In order to improve the results of the approaches mentioned above, and since we our handling French text data for which the usual tools and libraries are less finely-tuned than for English, we

[3] Mikolov, Tomas, et al. « Efficient Estimation of Word Representations in Vector Space ». arXiv:1301.3781 [cs], 3, septembre 2013. arXiv.org, http://arxiv.org/abs/1301.3781.

try to build our own features: inspired by the tf-idf approach, we turn to what we call 'dictionary-based approaches'. The concept is simple: for every decision we have to make, we use a different dictionary. We give the details and results below.

### 3.2.1 Dictionary-based sex-classification

We build on the naive approach of the baseline which was counting, for every text, all the male/female words in all of the text. But we do it only in the vicinity of the accident context. We figured that around the accident, the pronouns were more likely to refer to the victim than in the whole text. And indeed we achieve better accuracy using this first trick. However, that requires us to pinpoint the location of the accident context. We do this by using the words around the accident dates of all texts in the training set and creating a dictionary of all those words. The higher the score of these words on the training phase, the higher the weight conferred to the male/female pronouns found in their vicinity in the testing and validation phases. The second trick, more natural, is to simply lump all male texts of the training set together, and all female texts too, and create separate dictionaries for both. Then, we look for words that either appear in one corpus but not in the other, or that have a significantly higher chance of occurring in one corpus than in the other. To further exploit the tf-idf notions, a final feature is created by testing the robustness of a word: the count of how many different texts contain it, as opposed to the count of how many times it occurs. We combine those features, along with the naive one of the baseline, and use a logistic regression to classify sex: this yields a 97% accuracy (up to 98% for some random states).

### 3.2.2 Dictionary-based accident-classification

As mentioned before, we are now in possession of an accident dictionary, tracking words that are frequently around the accident date. We use this dictionary to score dates found in the text. Specifically, we use a dictionary of words found just before the date, and a dictionary of words found just after it (7+7 words). This simple scoring algorithm yields a 72% accuracy on validation.

### 3.2.3 Dictionary-based consolidation-classification

For consolidation, an additional dictionary is used, one tracking the death of the victim. Some words are more common in texts mentioning the death of the victim. This dictionary scores a whole text and a threshold is used to determine whether to return 'n.a.' or not. The second step of the algorithm resembles the one used in the previous section: we use a width of 5 words before and after the consolidation dates of the training set and score the dates of texts in the testing set accordingly. This method yields 66% accuracy.

## 3.3 Comments on these approaches

Even though we managed to beat the baseline with these classical approaches, their major drawback is that they do not consider the relationships between the tokens of the context around the dates. Indeed, a succession of words may refer to an accident context but we would miss it by considering these keywords based approaches. For instance : `reconnaissance des maladies professionnelles ne retienne a compter du 31 mai 1994` may not be recognized as an accident context because there is no typical accident keyword such as `victime`, `accident`, `prejudice`. But the succession `reconnaissance des maladies professionnelles` shows that this sentence is talking about the date where we considered that the professional illness began (thus it is the accident date).

Furthermore, we did not tackle the data imbalance of the built contexts yet, therefore there is still some room for improvement.

Lastly, a big number of hyper-parameters are appearing and we cannot possibly hope to fine-tune them all (thresholds, widths of windows, scoring procedures, etc.) However, a neural network can.

# 4 Neural network approaches

## 4.1 BERT overview

In this section, we introduce BERT, which stands for Bidirectional Encoder Representations from Transformers[2]. But first, We should start by understanding how a Transformer works and what self-attention is. [1] since BERT is just the encoder part of a transformer : a succession of 12 encoders.

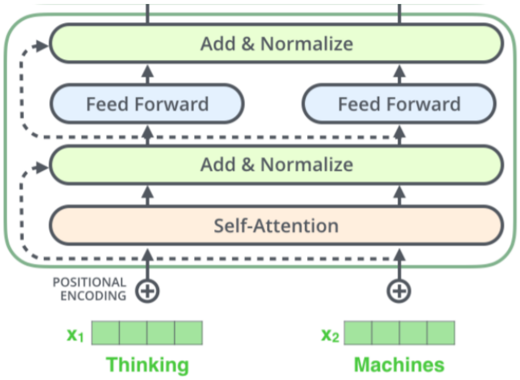


Figure 5: Constituents of an encoder stack

The encoder's inputs first flow through a self-attention layer – a layer that helps the encoder look at other words in the input sentence as it encodes a specific word which leads to a better encoding for this word. This is the major contribution and that differs from the classical RNN and Bi-RNN since with this attention mechanism, the encoder is "non-directional" (more detail on the attention mechanism can be found in the cited paper [1]). The outputs of the self-attention layer are fed to a feed-forward neural network. The exact same feed-forward network is independently applied to each position.

As mentioned before, BERT uses the encoder part of the Transformer to encode the sentence given as an input. The training of this encoding is performed on two different tasks (minimizing the sum of two different losses, each corresponding to a task) :

- Predicting a masked word from the input sentence : Analogous to any standard word embedding learning task, the goal is to find a representation of a word as a vector. The attention mechanism serves the goal of training the net in a Bidirectional way, but in reality the sentences in the input are being fed all at the same time, thus the real mechanism is "un-directional / non-directional" and the word being encoded pays attention to all the other words in the sequence.

- Prediction; if sentence A and B are two successive sentences : The goal of this is to empower its ability to generate coherent sentences and learn to catch "far" dependencies in the sentences.

In the end of this training, we obtain word embeddings that are **context dependent**. But not only that, since we can use other outputs of the model -such as the Start of sentence (SOS) token embedding- to perform another downstream task (in our case, a multi-class classification of the context) since this token encodes a sort of summary of the whole input sentence.

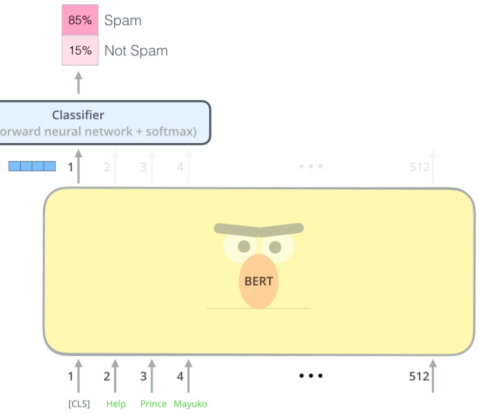


Figure 6: Constituents of and encoder stack

To sum it up, BERT (the Transformer's encoder) gives token embeddings that are very rich and efficient since they are context dependant, work for very long sentences and catch more types of dependencies between words in the sentence.

We use the HuggingFace library that implements CamemBERT : a version of BERT pretrained on a French corpus[4].

## 4.2 Class imbalance

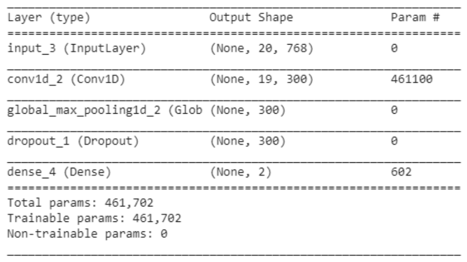
To handle the class imbalance for all the following models, we use a cost sensitive loss that we minimize.

$$L(y_{true}, y_{pred}) = \sum_{Ci} Cost_i * y_{true,i} * \log(\mathbf{P}(\mathbf{y_{pred,i}}))$$

## 4.3 Sex identifier

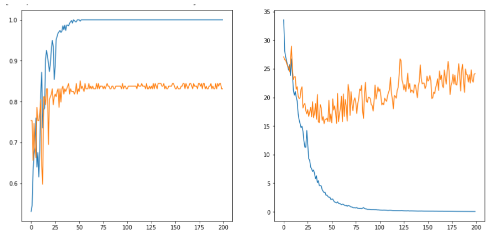
Given the embedding of the SOS tokens of the rows having the dates in a text (limiting it to 20 rows maximum for computational efficiency), we use a neural network (NN) similar to the ones that perform well for sentiment analysis predicting "homme" (class 0) or "femme" (class 1) with a cost sensitive loss. The cost we fixed is $Cost = [30, 70]$.

The performance on the validation set is not that great in comparison with the 'handmade' solution, but we saw that there can be a room for improvement also using this approach. (Plus, we wanted to try a fully NN approach to see how good it performs for the sake of comparison with the classical approach)



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_3 (InputLayer)         (None, 20, 768)           0         
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 19, 300)           461100    
_________________________________________________________________
global_max_pooling1d_2 (Glob (None, 300)               0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 300)               0         
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 602       
=================================================================
Total params: 461,702
Trainable params: 461,702
Non-trainable params: 0
_________________________________________________________________
```

(a) Architecture

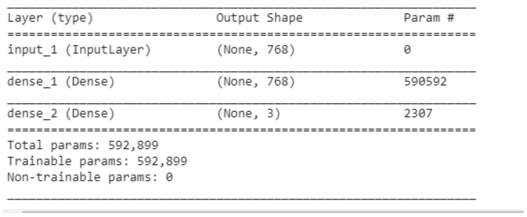


(b) Accuracy : train(Blue) | validation (orange)

## 4.4 Dates identifier

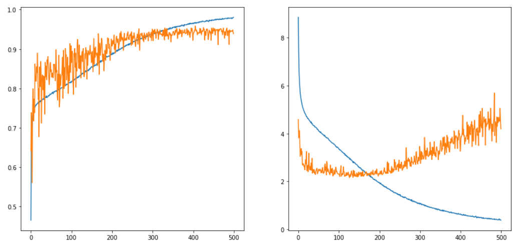
Same as for the classical approach, the goal here is to classify each context around a date as being an `accident` (Class 0), `consolidation` (Class 1) or `other` (Class 2).

For our case, we have chosen : $Cost = [48, 48, 4]$ giving importance to classifying with high probability the minority classes.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 768)               0
_________________________________________________________________
dense_1 (Dense)              (None, 768)               590592
_________________________________________________________________
dense_2 (Dense)              (None, 3)                 2307
=================================================================
Total params: 592,899
Trainable params: 592,899
Non-trainable params: 0
_________________________________________________________________
```

(a) Architecture



(b) Accuracy : train(Blue) | validation (orange)

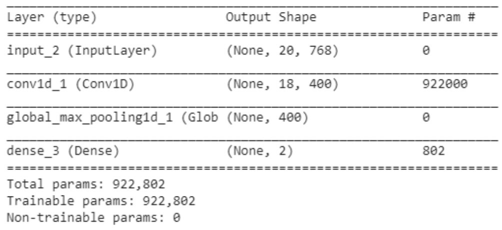
After training the model on the previously imbalanced built data set, we see that its validation accuracy exceeds 90%. But this is not the real performance of the model since we score sentences independently. The real performance is assessed starting from a text, processing it, building the contexts around them, building its BERT features then predicting the classes probabilities using the previously defined NN.

#### 4.4.1 Accidents date identifier

We predict as an accident date the one that has the highest accident probability that is over a fixed threshold (0.7 in our case ) else we predict "n.c.".

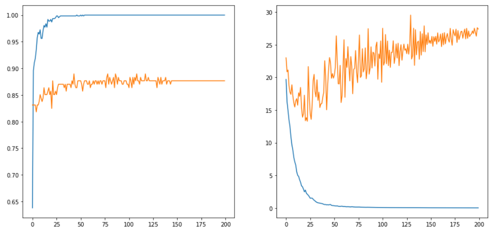
#### 4.4.2 Consolidation date identifier

At this step, we filter out all the dates that precede the accident date predicted, and look at the predicted probabilities for the consolidation. If it over a fixed threshold (0.75 in our case ), we predict the corresponding date, else we use a NN that is trained to predict if a context is a "death" context, for which we predict "n.a.", else we predict "n.c.". The NN used for this task is exactly similar to the one used for the NN sex prediction.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         (None, 20, 768)           0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 18, 400)           922000
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 400)               0
_________________________________________________________________
dense_3 (Dense)              (None, 2)                 802
=================================================================
Total params: 922,802
Trainable params: 922,802
Non-trainable params: 0
```

(a) Architecture



(b) Accuracy : train(Blue) | validation (orange)

## 5 Summary of the performances

| Approach | Accident | Consolidation |
|---|---|---|
| 2 classes - avg Word vectors | 49 | 41 |
| 2 classes - tf idf ** | 59 | 50 |
| 3 classes - avg Word vectors | 54 | 38 |
| 3 classes - tf idf ** | 65 | 44 |
| 3 classes - BERT - NN* | **78** | **67** |
| Dictionary | 72 | 66 |

Table 1: Validation accuracy of dates classifiers | * : using the constraint on the date consolidation | ** filtering some dates using a dictionary

| Approach | Sex |
|---|---|
| Naive | 88 |
| Dictionary | **98** |
| BERT 20 rows + Conv1D NN | 82 |

Table 2: Validation accuracy of sex prediction

# 6 Why is BERT powerful ?

As explained earlier, the core component of BERT is the attention heads that serve the sake of computing the attention weights on the tokens of the sentence. This is what we are going to analyze here.

Roughly speaking, attention is a way for a model to assign weight to input features based on their importance to some task. When deciding whether an image contains a dog or cat, for example, a model might pay more attention to — i.e. place more weight on — the furry parts of the image as opposed to the lamp or window in the background. Similarly, a language model that is trying to complete the sentence `lausanne tenus indemniser integralite prejudice subi monsieur y...suite ____ survenu 11 septembre 1997` may want to pay more attention to the word `monsieur`, `prejudice` and `subi` than `lausanne`, because knowing that the subject is "monsieur Y" is more important for predicting the next word "accident" than knowing where the accident happened (here lausanne remained due to our pre-processing truncation).

In Figure 10 we plot as a heat map, the attention weights of each attention head of the last layer of the CamemBERT's encoder. First thing to notice is that the SOS token (the one whose embedding we use for the classification) has important weights towards the other words of the sentence, and this for all the attention heads, which confirms that the SOS token embedding encodes in it a little bit of information coming from every token of the sentence.

Attention can also be used to form connections between words, enabling BERT to learn a variety of rich lexical relationships. This is what the attention heads 0 and 3 of the previous graph (represented below) show. Both heads give important weights for the terms `victime`, `accident`, `consequences` and `dommageables`.

All these components serve the purpose of giving rich 768-dimensional features that are used to classify the contexts. This complex relationship between the tokens is the prior information that we did not manage to include in the classical approaches. This lack of inter-dependency along with the fact that the mean of the word vectors is not a good summary of the context we are processing are the reasons why BERT outperforms them on the public leader-board with a gain of 7 to 12 points relatively to a fully classical approach based on dictionaries, keywords and their closeness to the date detected. In addition, there was no need to force the filtering of some rows (of the court, laws, birthdays...) before scoring the contexts with our classifier and this is very useful since it removes an extra parameter (list of terms to that discard the context from being scored) and prevents errors too (for example in the case where an accident date is in the same context as a word we included in that discarding terms list)

# 7 Limits of our models

## 7.1 Classical approaches

For the Tf-idf approaches as well as the dictionary based approaches, the main fundamental limitation is that they do not work for sentences that do not contain the keywords needed for the clas-

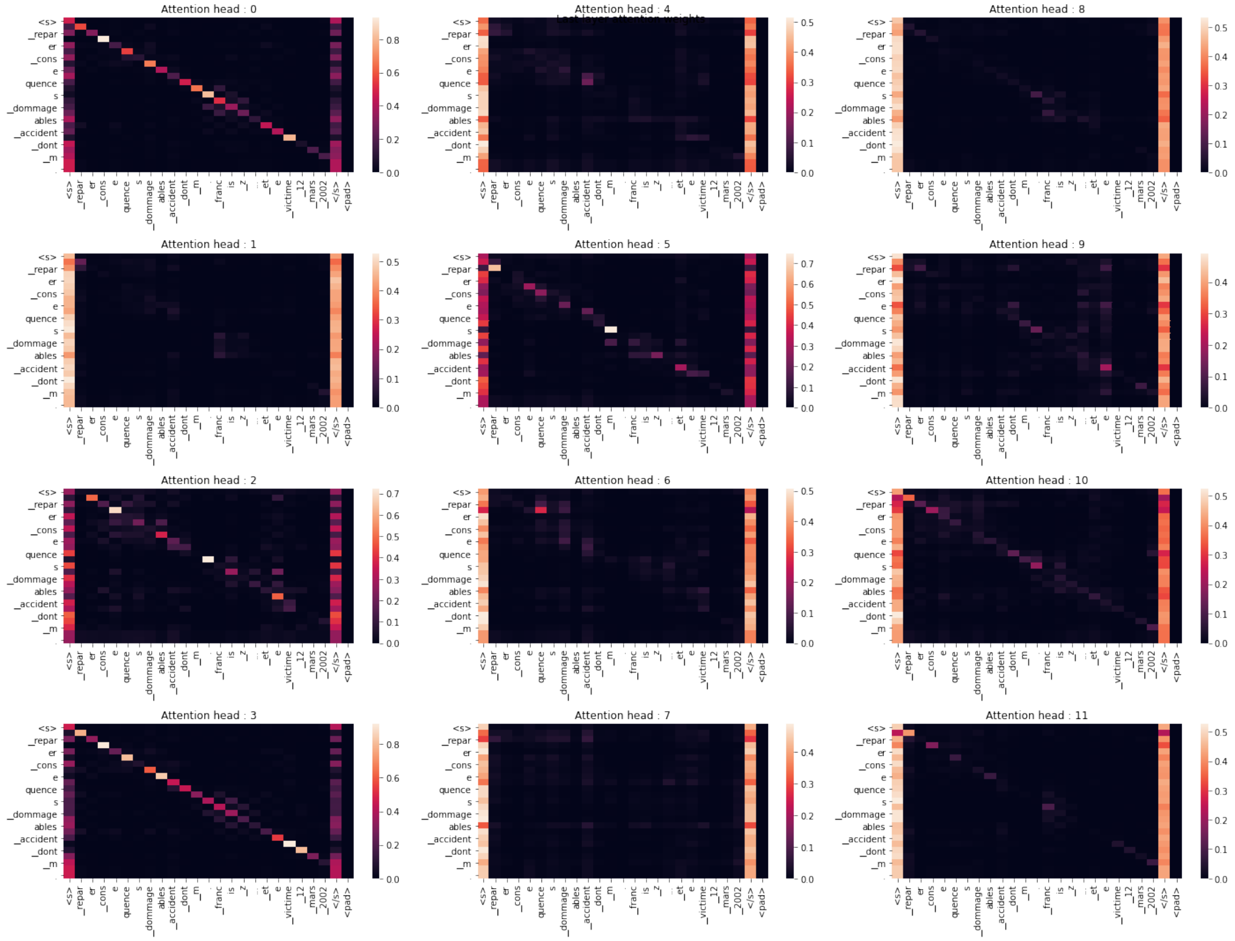


Figure 10: Last encoder's layer attention weights of the sentence `'reparer consequences dommageables accident dont m. francis z... ete victime 12 mars 2002 '`

| Submission | Validation | Public LB |
|---|---|---|
| Naive sex classifier + 2 classes tf idf (conso) + 3 classes tf idf (accident) | 65 | 71 |
| Dictionary sex classifier + 2 classes tf idf (conso) + 3 classes tf idf (accident) | 71 | 73 |
| NN sex classifier + 3 classes BERT NN (constrained) | 75 | 79 |
| Dictionary sex classifier + 3 classes BERT NN (constrained) | 77 | 82 |

Table 3: Summary of the successful submissions

sification. This is not a problem for the accident context since in the majority of cases we can find those words, but the problem happens for the consolidation date ( as we can see on the validation accuracy of those models for the consolidation date).

### 7.2 The dependency between predictions

We base the predictions of the consolidation date on the prediction of the accident date with the constraint of date accident < date consolidation. This assumes that our accident context scorer to be very accurate in its predictions (giving high probabilities for contexts of accidents). This is why we tried fitting two independent classifiers consisting of 2 classes predictors (Accident/not accident - consolidation / not consolidation) but the performance was not that great on the validation.

### 7.3 Pre-processing

The other technical problem is the date formats in the original texts. Indeed, the first step we take is uniforming the dates, and this is a method based on regular expressions where we list the different formats of dates that we encountered in the training texts so as to parse them. This list is not exhaustive and one may find another format of date that is not listed, thus we will not manage to find the date in text. Therefore the corresponding context might not even be in the list of candidates to be scored. This is less problematic for the sex prediction using the neural classifier ( where we perform 1D convolution on the SOS tokens embedding of rows containing dates). With our last date processor (date_unifomizer.py), the problem was persisting on the dates that are mentioned in a time span : `Du 03.06 au 07.06.2003 a été hospitalisé due à....`, here the accident date is `3 juin 2003` but we do not even parse it with our actual processing.

### 7.4 Over-fitting

Unlike Kaggle, we did not find how to get the private score of all of our submissions, so the only private score we got was the one of the submission using the dictionary based sex classifier and the Neural networks fit on BERT features constrained with the order of date accident and consolidation along with the Convolutional network to predict 'n.a.' or 'n.c.' for the consolidation date.

This approach scored 82% accuracy on the

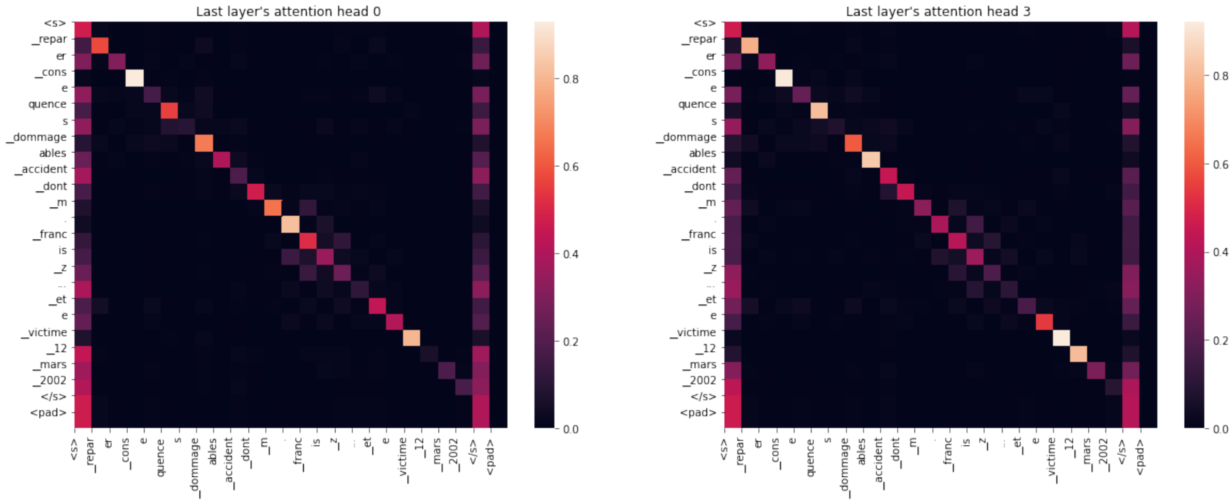


Figure 11: Important attention weights for keywords

public leader-board but only 77% on the private leader-board. We believe that this over-fitting came from :

- Fixing hyper-parameters using training set only : Such as the maximum number of tokens to consider before and after the date $T$ and threshold of probabilities (under which we predict n.c or n.a).
- The hypothesis concerning the occurrence of dates of interest coming after the first mention of the clerk in the text.
- Some new format of dates that our date uniformizer does not manage to parse (even though we believe that this might not be very impacting and only concerns a few texts)

## 8 Conclusions and further work

The fact that a pre-trained model on a French corpus (subcorpus of the newly available multilingual corpus OSCAR) unrelated to legal and juridical documents manages to give a pretty good accuracy on such a task is quite surprising. One may argue that further fine-tuning the weights of the CamemBERT encoder on these legal documents will boost the performance by allowing the encoder to output features particularly built for this classification task. We attempted to do that, but we faced the problem of memory overflow (even using the Free Google Colab GPU servers) which forced us to work with the output of the pre-trained model.

We also showed that we should not always consider a neural network as a solution for everything especially when there is no need for it. Indeed with a little bit of effort dedicated to crafting hand-made features for sex prediction, we can achieve a nearly perfect accuracy (up to 98%). Another proof for this are the MFCC features in all that concerns audio processing - Although they are not 'handmade', we do not rely on a neural network to learn those complex features.[5]

# References

[1] Jay Alammar. In: (2018). URL: https://jalammar.github.io/illustrated-transformer/.

[2] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805.

[3] Vairaprakash Gurusamy and Subbu Kannan. "Preprocessing Techniques for Text Mining". In: Oct. 2014.

[4] Louis Martin et al. *CamemBERT: a Tasty French Language Model*. 2019. arXiv: 1911.03894 [cs.CL].

[5] Sirko Molau et al. "Computing Mel-Frequency Cepstral Coefficients On The Power Spectrum". In: (Feb. 2002).

[6] Octavia-Maria Sulea et al. "Exploring the Use of Text Classification in the Legal Domain". In: *CoRR* abs/1710.09306 (2017). arXiv: 1710.09306. URL: http://arxiv.org/abs/1710.09306.