Data Science Challenge

NLP applied to judicial decisions parsing

Sebastien McRae & Mohamed Ali Chaieb & Omar Kadim

March 13, 2020

CentraleSupélec

Predilex

Summary

Challenge Description

Challenge Context

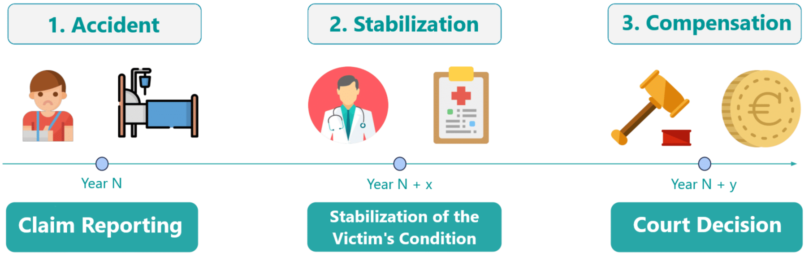

Figure 1: Journey of a victim of personal injury

Goals of the Challenge

Extraction, from case law (text) documents, of **3 features** :

- Victim's **gender**
- **Date of the accident**
  - In the format YYYY-MM-DD,
  - or "UNK." if the date is missing.
- **Date of stabilization** of the victim's condition
  - In the format YYYY-MM-DD,
  - or "UNK" if the date is missing,
  - or "N/A" if the victim died before stabilization.

Metric : Mean accuracy over the 3 features.

Some Examples (Accident Date)

- « Laetitia X. [...] during a foam party, on **August 1, 1998**, [...] was seriously injured after slipping on the dance floor of this nightclub. »
- « On **May 27 2009**, [...], Alexandre X. lost control of his vehicle due to a malaise. »
- « On **6 March 2009**, during a football training match [...], Jean-Ulrich X., born on 9 march 1994, suffered an injury. »

A few comments (Dates)

- Documents from various courts and judges (**inconsistent structure**).
- Sentences sampled over a period of time long enough to observe **formulation changes**.
- **Few data** in view of the diversity of the formulations used.

$\longrightarrow$ Idea: «Fine-tune» a pre-trained model that has the ability to capture the semantic context of dates.

The Benchmark

Gender Classification: counting the number of occurrences of key words such as "he" vs "she", "Mr." vs "Mrs.", ...

Date Extraction:

- Embedding of sentences that contain dates with *BagOfWords*.
- Ranking by distance to a SVM linear classifier.
- The prediction for each field is the date that had the best score, if this score was higher than a threshold (otherwise they predicted "UNK").

Accuracy: 0.6458

Data Preprocessing

Classification of the Victim's Gender

- **Deleting Special Characters** (punctuation marks and diacritics, miscellaneous symbols).
- **Removal of numeric characters** (not correlated with the victim's gender classification).
- **Stop words removal** (non-significant words that are uniformly distributed throughout the collection of texts : «the», «of», ...).

Labeling of Sentences that Contain Dates

The Issue :

- We have 3 labels per document but not the labels of the sentences of these documents.

- Several dates can be present in the same sentence.

- Very diverse date formats.

Context Around the Dates - 1/4

- We split the documents by sentences.
- **If the sentence only contains a date**, we label the whole sentence.
- **If the sentence contains several dates**, we split the context in 2 between the dates (with overlap if the size of the context is below a certain threshold).
- **In any case**, we keep at most K context tokens (K/2 on each side).

3 hyperparameters: The size of the context, the overlap threshold and the size of the overlap.

Context Around the Dates - 2/4

The professor concluded that the traffic accident that occurred on **October 23rd, 1998** **may have had a direct role in the vascular-cerebral accident on** **December 23rd, 1998**.

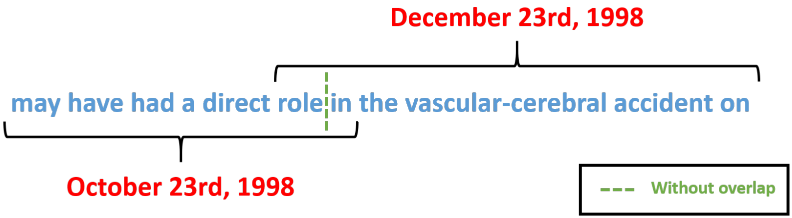

Figure 2: Illustration with overlap = 1

Context Around the Dates - 3/4

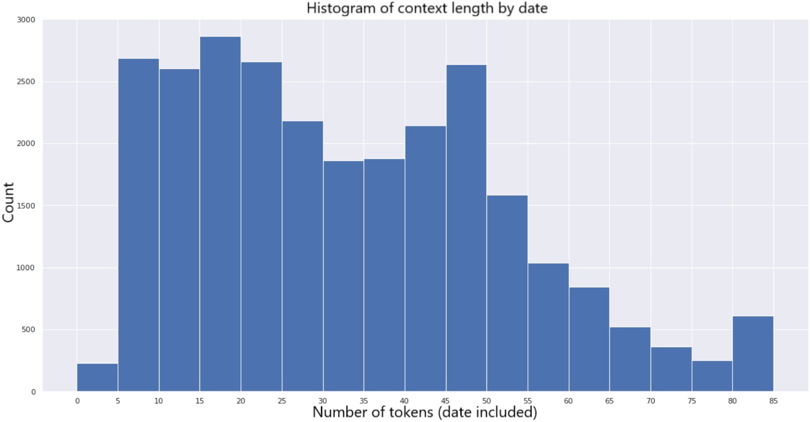

Figure 3: Histogram without overlap (K=80)

Context Around the Dates - 4/4

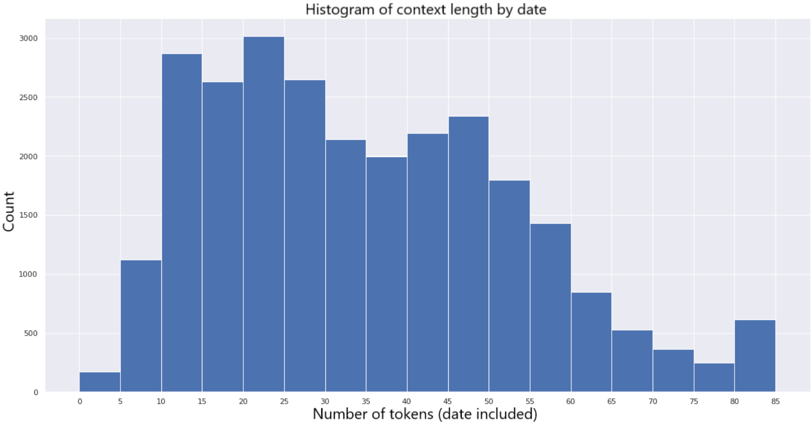

Figure 4: Histogram with overlap = 5 and a thredshold of 20 tokens (K=80)

Our Models

3 Models

- **Gender** (binary classification)
- Input : Text Document
- Output : "Male" or "Female"

- **Dates** (multi-class classification)
- Input : Sentence that contains a date
- Output : "None" or "Accident Date" or "Stabilization Date"

- **Non-stabilization of the victim's condition** (binary classification)
- Input : Text Document
- Output : "UNK" or "N/A"

Classification of the Victim's Gender - 1/4

Label distribution:

- "UNK": only 5 labels of undetermined sex, are not taken into account
- "Female": 206 / 770 labels
- "Male" : 559 / 770 labels

Taking into account the distribution of labels: Use of *classWeights* in the calculation of errors during training.

Classification of the Victim's Gender - 2/4

Architecture of the classification model :

- Transformation of each text in the training base into a list of tokens of 1000 words in length.
- Application of an *embedding* layer with dimension 100 followed by a *dropout* layer to avoid overfitting.
- Application of a 1-dimensional convolution layer *conv1D*, with a filter number = 250 and a size = 3 followed by a *GlobalMaxPooling* operation.
- Projecting the last layer of the network onto a neuron followed by a *sigmoid* activation that gives an output in [0,1].

Classification of the Victim's Gender - 3/4

Model training:

- Using the Adam optimizer with default settings.
- The cost function used is the binary cross-entropic loss adapted to a binary classification problem with precision metrics (*accuracy*).

Generalization: **split** coefficient = 0.2 and **validation accuracy** = 0.96

Classification of the Victim's Gender - 4/4

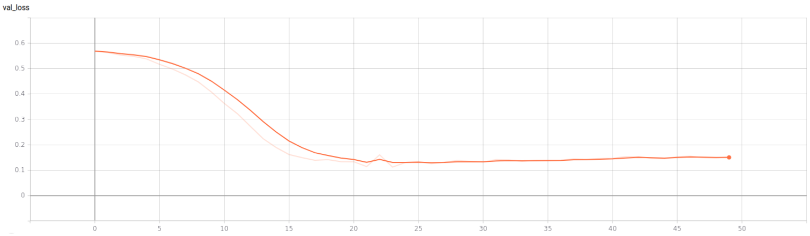

Figure 5: Evolution of the validation loss

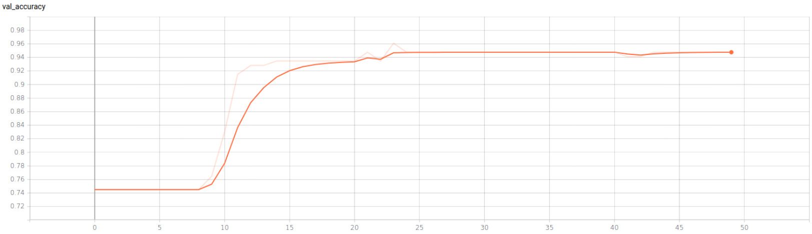

Figure 6: Evolution of validation accuracy

Date Classification

- Logits calculation with RoBERTa.
- The logits are weighted by the occurrences of the dates and a ranking is made.
- The date chosen for each date is the 1st in the ranking.

Distribution of Date Labels

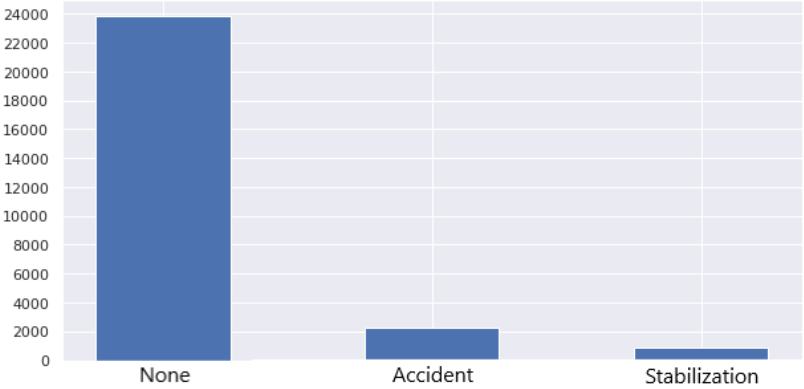

Figure 7: Train Set Histogram

Cost-Sensitive Loss Function

Balancing the cost function: The cost function is calculated using weights assigned to each class (due to the unbalanced distribution of training data).

Weights are calculated with the *Scikit-learn* function *compute_class_weights*.

$$L_{CS}\left(\mathbf{y}^{(k)}, \hat{\mathbf{y}}^{(k)}\right) = -\sum_{i} c_{ij}^{(k)} \left(y_i^{(k)} \log \hat{y}_i^{(k)}\right)$$

Detection of Non-stabilization of the Victim's Condition

Label distribution :

- "N/A": 102 / 770 labels
- Existing Date: 668 / 770 labels
- Use of a cost function sensitive to class imbalance in training data.

Model training: The same binary classification model defined for gender classification is used. The generalization accuracy of the test-set is about **0.93**.

Evaluation of our Models

Labelled Dataset: 770 documents.

For the dates: we remove the 3 documents that have the same date of accident and stabilization.

We split up our labeled dataset. :

- 80% of the documents for training purposes.
- 20% of the documents for test purposes.

For the dates: **Macro-F1** is used as a metric for classification at sentence level (advantage minority classes).

The Ranking

Ranking	Date	User(s)	Public score
1	March 5, 2020, 3:55 p.m.	omar-kadim & Mohamed-Ali-Chaieb & Sebduncan	0.9089
2	Feb. 28, 2020, 10:27 p.m.	dot & mathieu.rita	0.7760
3	Feb. 26, 2020, 10:39 a.m.	gitting_guud & marvinkaze	0.7292

Figure 8: The podium on March 13, 2020

State of the Art

State of the Art

Model	Error	Paper / Source	Code
XLNet (Yang et al., 2019)	0.62	XLNet: Generalized Autoregressive Pretraining for Language Understanding	Official
Bidirectional Encoder Representations from Transformers (Devlin et al., 2018)	0.64	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	Official
ULMFiT (Howard and Ruder, 2018)	0.80	Universal Language Model Fine-tuning for Text Classification	Official
CNN (Johnson and Zhang, 2016)	0.84	Supervised and Semi-Supervised Text Categorization using LSTM for Region Embeddings	Official
DPCNN (Johnson and Zhang, 2017)	0.88	Deep Pyramid Convolutional Neural Networks for Text Categorization	Official
VDCN (Alexis et al., 2016)	1.29	Very Deep Convolutional Networks for Text Classification	Non Official
Char-level CNN (Zhang et al., 2015)	1.55	Character-level Convolutional Networks for Text Classification	Non Official

Figure 9: State of the art in text classification (DBpedia)

Link : **github.com/sebastianruder/NLP-progress**

About RNNs

- They are recurrent, the time dependence of the cells limits the parallelization.

- *Vanishing gradient* problems.

- LSTM: Each cell receives as an input the previous *hidden state* and the *cell state*

The Attention Principle

History :

- Used by Google DeepMind for **image classification** in 2014 in «Recurrent Models of Visual Attention». [1]
- Then used in 2015/2016 for machine translation. **NLP** tasks based on **RNN/CNN** in «Neural Machine Translation by Jointly Learning to Align and Translate». [2]
- In 2017, Google made extensive use of self-attention mechanisms to learn **textual representation** in the article «Attention is all you need». [3]

Transformers

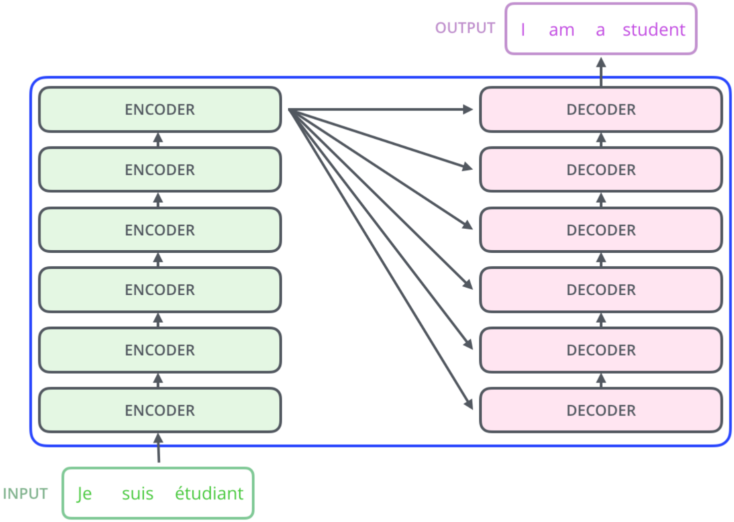

Figure 10: Encoder-decoder architecture of a transformer [4]

The Encoder

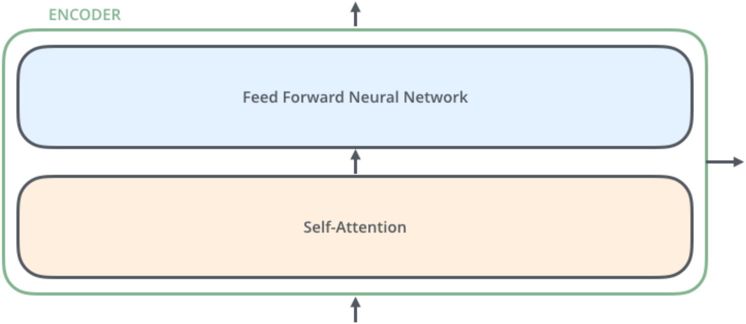

Figure 11: One of the 6 encodeurs [4]

The Encoder in Detail

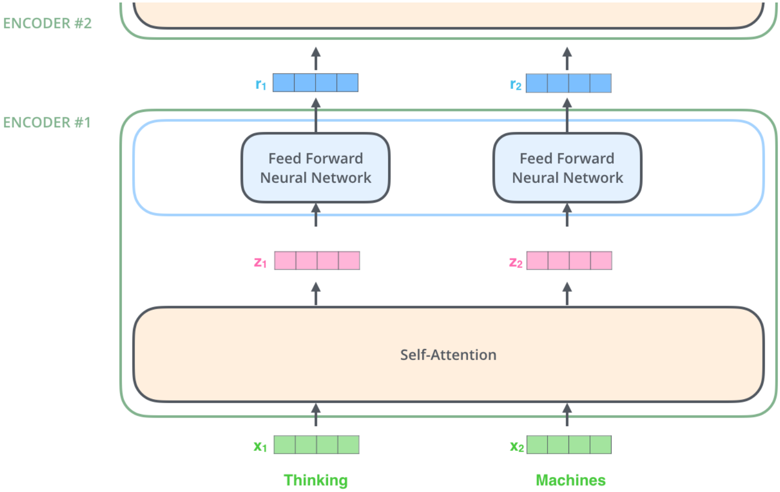

Figure 12: Internal architecture of an encoder [4]

The Attention Principle

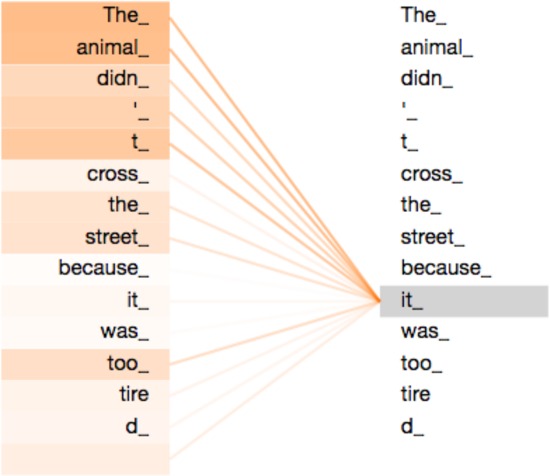

Figure 13: Illustration of the attention principle [4]

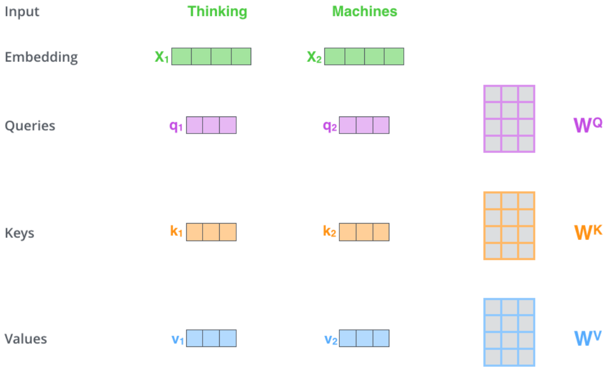

Figure 14: Introduction of the 3 vector representations of each word [4]

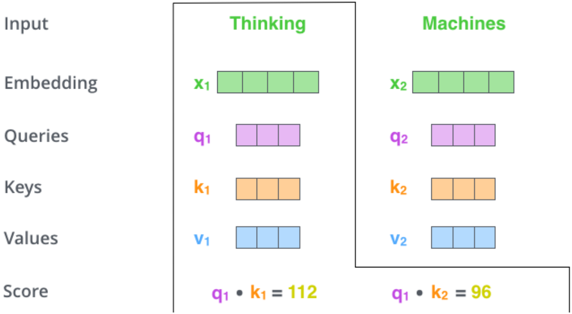

Figure 15: Computation of the self-attention [4]

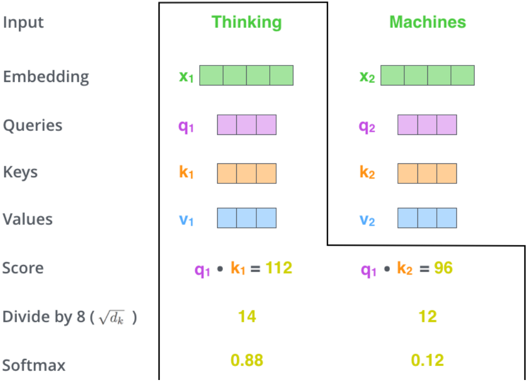

Figure 16: Computation of the self-attention [4]

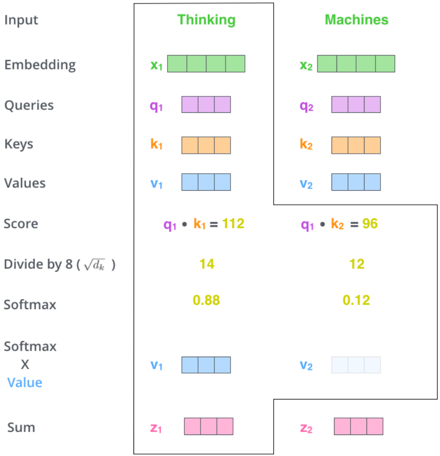

Figure 17: Computation of the self-attention [4]

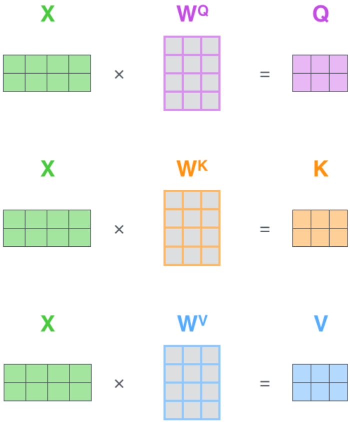

Figure 18: Queries, Keys and Values in matrix form [4]

The Attention Principle in Detail - 6/11

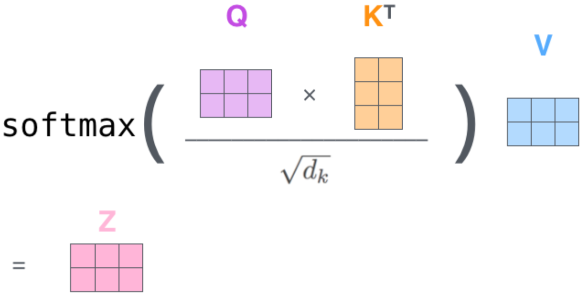

Figure 19: Calculation of the output of the self-attention layer [4]

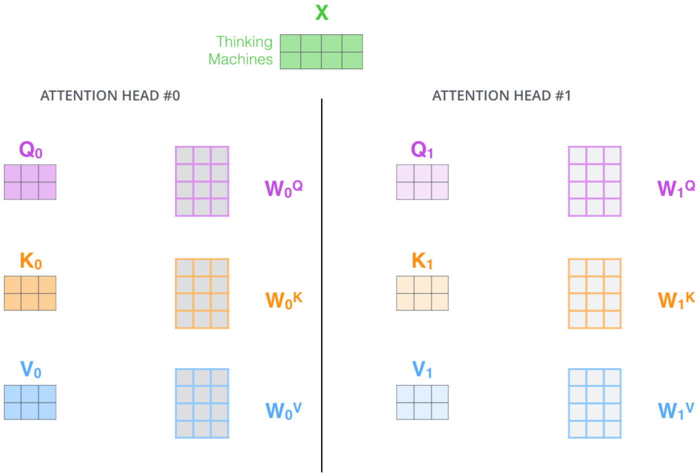

Figure 20: Illustration of the principle of multi-headed attention [4]

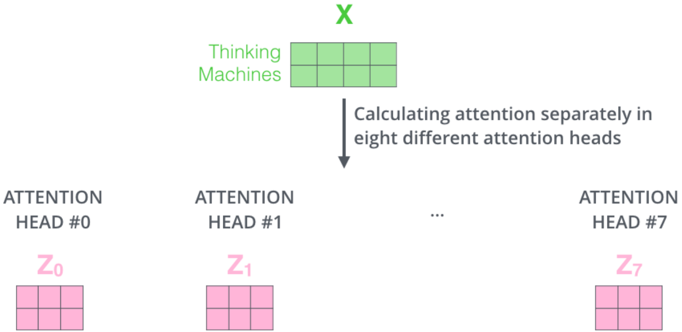

Figure 21: Calculation of the output of the attention heads [4]

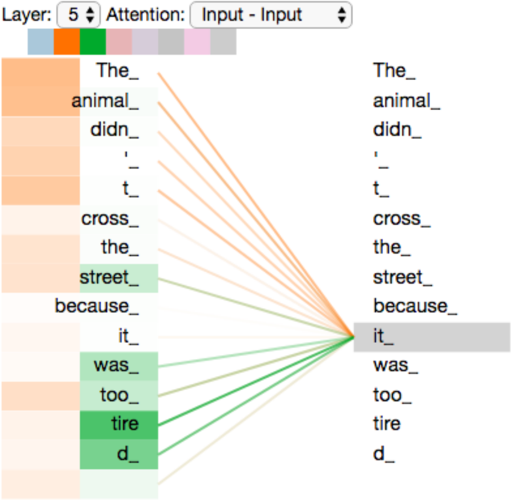

Figure 22: As we encode the word "it", one attention head is focusing most on "the animal", while another is focusing on "tired" [4]

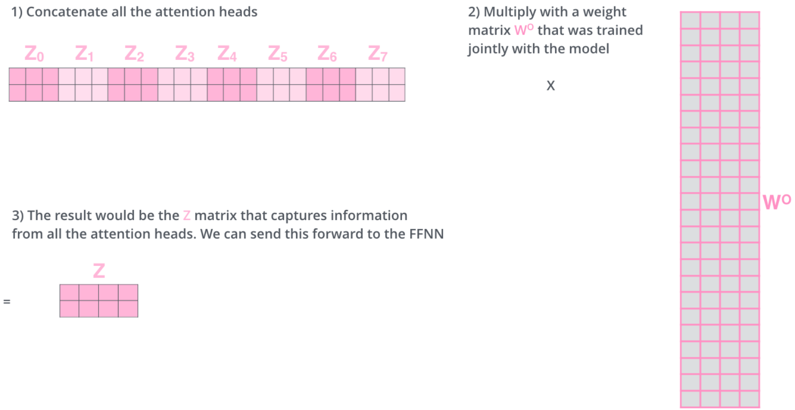

Figure 23: Concatenation of the output of the attention heads [4]

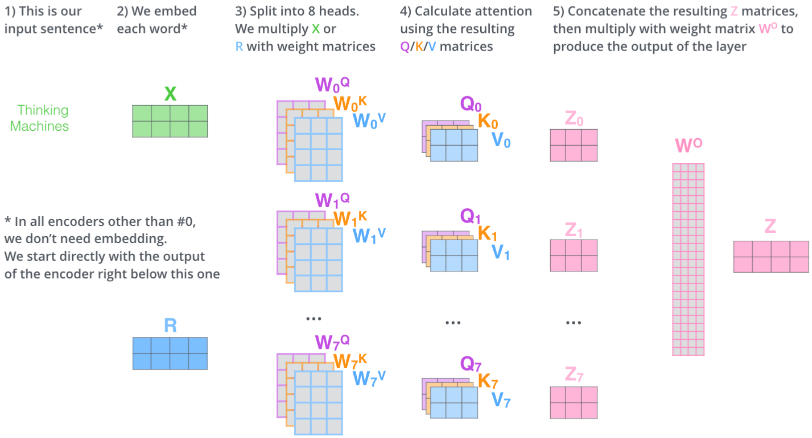

Figure 24: Recap of the Attention Principle [4]

Positional Encoding of Words

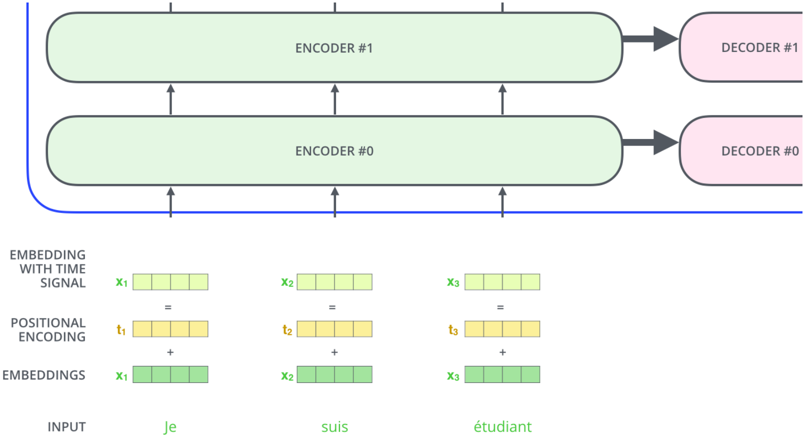

Figure 25: Embedding Vectors [4]

Positional Encoding of Words

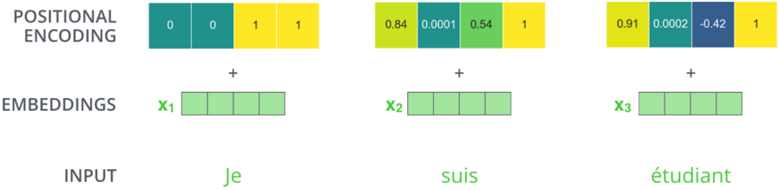

Figure 26: An example of positional encoding of size 4 [4]

The Encoder in Detail

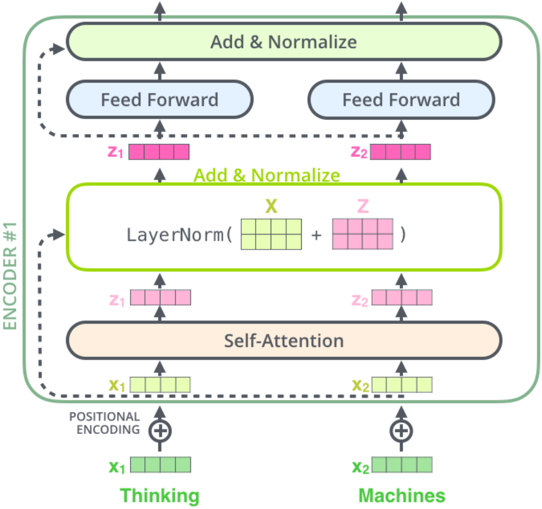

Figure 27: Full Encoder Architecture [4]

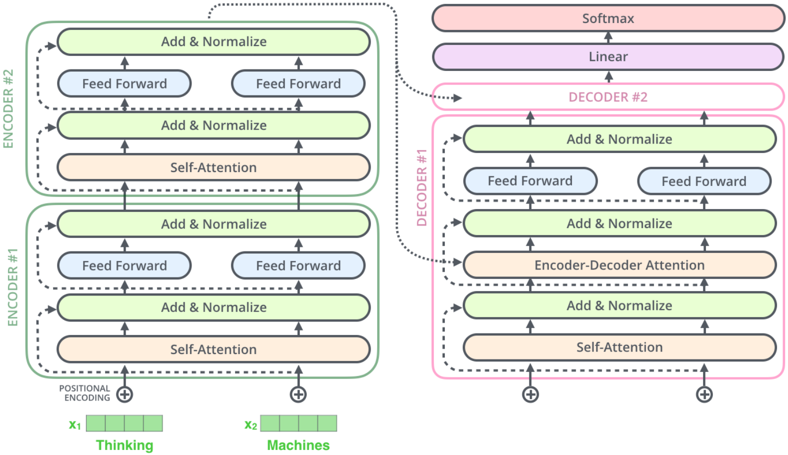

Figure 28: The Partial Architecture of a Transformer [4]

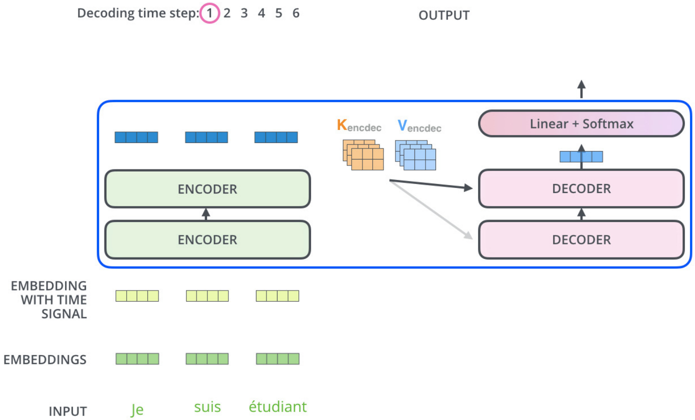

Figure 29: Decoder Illustration [4]

The Decoder - 2/2

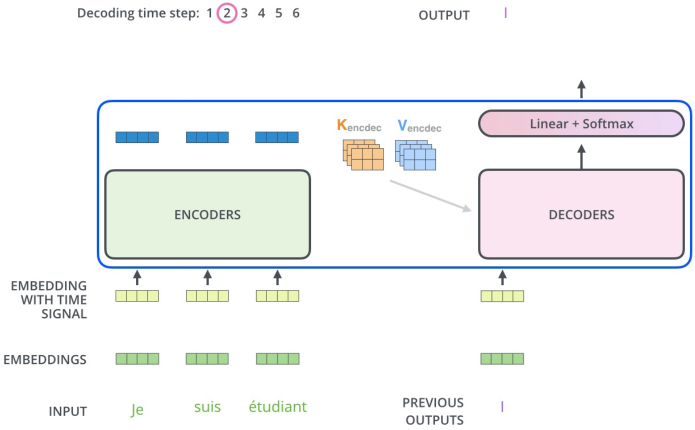

Figure 30: Decoder Illustration [4]

The Output of the Transformer

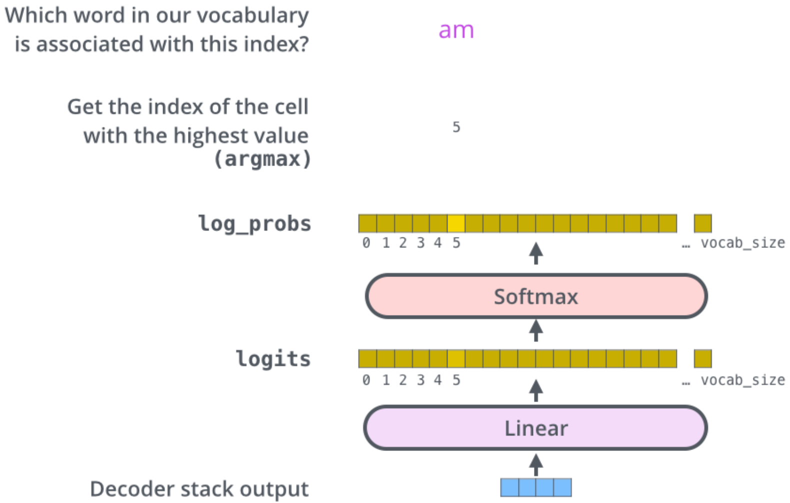

Figure 31: Transformation of the decoder output into a word

Training

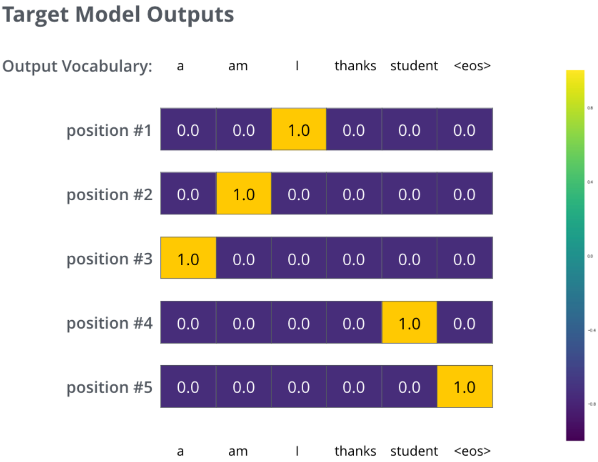

Figure 32: Target Probability Distributions [4]

Training

Trained Model Outputs

Output Vocabulary:	a	am	I	thanks	student	<eos>
position #1	0.01	0.02	0.93	0.01	0.03	0.01
position #2	0.01	0.8	0.1	0.05	0.01	0.03
position #3	0.99	0.001	0.001	0.001	0.002	0.001
position #4	0.001	0.002	0.001	0.02	0.94	0.01
position #5	0.01	0.01	0.001	0.001	0.001	0.98
	a	am	I	thanks	student	<eos>

Figure 33: Post-training Probability Distributions [4]

Classification with BERT

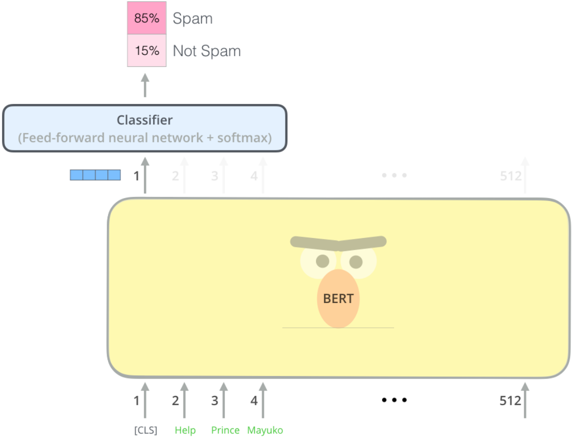

Figure 34: BERT «Fine-tuning» for a classification task [4]

Appendices

Appendix: F1 Scores

Weighted-F1 $\propto F1_{classe1} * W_1 + F1_{classe2} * W_2 + \cdots + F1_{classeN} * W_N$

The F1 score is calculated for each class independently, but weighted by the occurrences of each class: **thus favoring the majority ranking**.

Micro-F1 $= F1_{classe1+classe2+\cdots+classeN}$

We use the global number of TP, FN, FP and calculate directly the F1 score: **does not favor any particular class**.

Macro-F1 $\propto F1_{classe1} + F1_{classe2} + \cdots + F1_{classeN}$

The F1 score is calculated separately but without using weights for aggregation: **greater penalty when the model does not work well with the minority classes**.

[1] Volodymyr Mnih et al. "Recurrent Models of Visual Attention". In: *CoRR* abs/1406.6247 (2014). arXiv: 1406.6247. url: http://arxiv.org/abs/1406.6247.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2014. url: http://arxiv.org/abs/1409.0473.

[3] Ashish Vaswani et al. "Attention Is All You Need". In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. url: http://arxiv.org/abs/1706.03762.

[4] Jay Alammar. "http://jalammar.github.io/illustrated-transformer/". In: (Jan. 2018).

[5] 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.

[6] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *CoRR* abs/1907.11692 (2019). arXiv: 1907.11692. url: http://arxiv.org/abs/1907.11692.

[7] Louis Martin et al. "CamemBERT: a Tasty French Language Model". Web site: https://camembert-model.fr. Oct. 2019. url: https://hal.inria.fr/hal-02445946.