Sequence Labelling & Classification

Machine Learning for Natural Language Processing, ENSAE 2022

Lecture 5

Benjamin Muller, INRIA Paris

Lectures Outline

1. The Basics of Natural Language Processing (February 1st)
2. Representing Text with Vectors (February 1st)
3. Deep Learning Methods for NLP (February 8th)
4. Language Modeling (February 8th)
5. **Sequence Labelling (Sequence Classification) (February 15th)**
6. Sequence Generation Tasks (February 15th)

Framework & Outline

We assume an input sequence of tokens $(x_1, .., x_T) \in V^T$.

We want classify each element in the sequence with the label $(y_1, .., y_T) \in [|1, L|]^T$.

Our goal is to estimate (Sequence Labeling)

$$\boxed{p_\theta(y_1, .., y_T | x_1, .., x_T)}$$

Framework & Outline

We assume an input sequence of tokens $(x_1, .., x_T) \in V^T$.

We want classify each element in the sequence with the label $(y_1, .., y_T) \in [|1, L|]^T$.

Our goal is to estimate (Sequence Labeling)

$$\boxed{p_\theta(y_1, .., y_T | x_1, .., x_T)}$$

For sequence classification, we simply consider y_T only

Framework & Outline

We assume an input sequence of tokens $(x_1, .., x_T) \in V^T$.

We want classify each element in the sequence with the label $(y_1, .., y_T) \in [|1, L|]^T$.

Our goal is to estimate

$$\boxed{p_\theta(y_1, .., y_T | x_1, .., x_T)}$$

Outline

1. NLP tasks
2. How to model them with Deep Learning?

Sequence Labeling & Classification Examples

- **Part-of-Speech Tagging**
- **Named Entity Recognition**
- The GLUE/SuperGlue Benchmark: **Boolean QA**
- **Hate Speech Detection**

POS Tagging

- **Input: Sequence of words (i.e. word-level tokenization is assumed)**
- **Output: For each word, predict the grammatical category**

Why doing POS tagging?

- Linguistic Analysis of a given corpus of text (**Sociolinguistics, Historical Linguistics…**)
- Language Acquisition Application
- Measuring the ability of a given **NLP technique**

What POS Tagset?

Defining all the possible **grammatical category** of a word depends on

1. What **language** you are working with?
2. A given **theory of syntax**

What POS Tagset?

Defining all the possible **grammatical category** of a word depends on

1. What **language** you are working with?
2. A given **theory of syntax**

Consequences:

➔ There is **no truly universal tagset** that would work in every cases

Still

- There is a ***Universal Dependency Corpora*** which attempts to do so

Universal Dependency Project (UD)

- Universal Dependencies (UD) is a framework for consistent annotation of grammar: **parts of speech, morphological features, and syntactic dependencies**
- Across **100 human languages**
- That produced so far about 200 Treebanks

Universal Dependency Project (UD)

- Universal Dependencies (UD) is a framework for consistent annotation of grammar: **parts of speech, morphological features, and syntactic dependencies**
- Across **100 human languages**
- That produced so far about 200 Treebanks

```
1	They	they	PRON	PRP	Case=Nom|Number=Plur	2	nsubj	_	_
2	buy	buy	VERB	VBP	Number=Plur|Person=3|Tense=Pres	0	root	_
3	and	and	CONJ	CC	_	2	cc	_	_
4	sell	sell	VERB	VBP	Number=Plur|Person=3|Tense=Pres	2	conj	_
5	books	book	NOUN	NNS	Number=Plur	2	dobj	_	SpaceAfter=No
6	.	.	PUNCT	.	_	2	punct	_	_
```

Universal Dependency Project: Tagset

17 POS Categories

Example:

He	*PRON*
owns	*VERB*
a	*DET*
house	*NOUN*
in	*ADP*
Paris	*PROPN*

- ADJ: adjective
- ADP: adposition
- ADV: adverb
- AUX: auxiliary
- CCONJ: coordinating conjunction
- DET: determiner
- INTJ: interjection
- NOUN: noun
- NUM: numeral
- PART: particle
- PRON: pronoun
- PROPN: proper noun
- PUNCT: punctuation
- SCONJ: subordinating conjunction
- SYM: symbol
- VERB: verb
- X: other

Universal Dependency Project: Tagset

Open class words	Closed class words	Other
ADJ	ADP	PUNCT
ADV	AUX	SYM
INTJ	CCONJ	X
NOUN	DET	
PROPN	NUM	
VERB	PART	
	PRON	
	SCONJ	

POS Tagging Evaluation

Accuracy of POS prediction over a test set of size N words:

$$Accuracy = \frac{\#\{y_i = \hat{y}_i\}}{N}$$

POS Tagging Evaluation

Accuracy of POS prediction over a test set of size N words:

$$Accuracy = \frac{\#\{y_i = \hat{y}_i\}}{N}$$

NB: This accuracy assumes “gold” word-level tokenization

Is POS a hard task?

- For ***high-resource languages*** we are near **99% accuracy** e.g. Camembert reached **+98% accuracy on French**

- For ***low-resource languages***: it is much harder ~50% for **Kurmanji** (Kurdish language)

NER

Def: ***NER consists in identifying the Name Entities in a sentence.***

For instance, we may want to identify:

PERSONS, LOCATION and ORGANISATION

United Nations official heads for Baghdad

➔ [ORG United Nations] official [PER Ekeus] heads for [LOC Baghdad]

We frame this task as a word-level sequence labelling task

NER

To do so, we can use a BIO approach (Beginning-Inside-Outside)

United	B-ORG
Nations	I-ORG
official	O
Ekeus	I-PER
heads	O
for	O
Baghdad	I-LOC

NER Evaluation

$$F1 = hmean(precision, recall) = \frac{2}{\frac{1}{precision} + \frac{1}{recall}}$$

Precision: *% of named entities that are correct out of the total number of predicted entities by the system*

Recall: *% of named entities that are correct out of the total number of name entities in the dataset*

GLUE / SUPERGLUE Benchmarks

The General Language Understanding Evaluation (GLUE) benchmark is a collection of resources for training, evaluating, and analyzing natural language understanding systems. GLUE consists of 9 tasks

Example: ***Bool QA*** predict **YES/NO** Given a question and a passage
We can frame it as a sequence classification task after concatenating the question and the passage

Sample

Question: "is france the same timezone as the uk",
Passage: "At the Liberation of France in the summer of 1944, Metropolitan France kept GMT+2 as it was the time then used by the Allies (British Double Summer Time). In the winter of 1944--1945, Metropolitan France switched to GMT+1, same as in the United Kingdom, and switched again to GMT+2 in April 1945….
Answer : false

Modeling for Sequence Labeling

Modeling

- **Sequence Labeling with LSTM-based model**
- **Sequence Labeling with a Transformer model**

RNN for Sequence Labeling

We assume an input sequence of tokens $(x_1, .., x_T) \in V^T$.

We want classify each element in the sequence with the label $(y_1, .., y_T) \in [|1, L|]^T$.

$$h_{i+1,t+1} = RNN_i(h_{i,t}, h_{i+1,t}), \forall\, i \in [|1, L|]\, \forall\, t \in [|1, T|]$$

$$\text{with } h_{1,t} = Emb(x_t) \text{ and } \hat{p_{t+1}} = h_{L+1,t+1}$$

$$\text{with } \varphi_L = softmax$$

- **So far, very close to language modeling**
- The main difference is that **we classify in a set of length *L***

RNN for Sequence Labeling

Limit: We model the sequence only **unidirectionally**

In ambiguous cases, **we need the entire sequence to predict the correct label:**

Example: ***st-gervais ski resort is an amazing place for skiing***

Impossible for a model to predict that *st-gervais ski resort* is a location without the right context

How to build a Bi-Directional DL Model?

Solution 1:

➔ Combine two RNNs, one for each direction (e.g. BI-LSTM)

Solution 2:

➔ Use a Transformer Model

Transformer for Sequence Labeling

Inputs: Transformers requires a fixed sequence at input (we note it $\mathcal{T}$)

Let's assume we have a sequence $(x_1, ...x_T)$

We simply append it with a ***PADDING*** token

$$\text{We append } (x_{T+1}, .., x_{\mathcal{T}}) \text{ with } \quad x_t = [PAD] \, \forall \, t \geq \, T + 1$$

We get a sequence of length $\mathcal{T}$: $(x_1, ...x_{\mathcal{T}})$

We make the model ignore those tokens by setting the softmax scores to 0 in the self-attention

Transformer for Sequence Labeling

Input

Embeddings:

$$(x_1, ...x_T)$$

Embedding: $(Emb(x_1), ...Emb(x_T))$

such that $Emb(x_i) = PositionEmb(x_i) + TokenEmb(x_i)$

Transformer for Sequence Labeling

Given a sequence of tokens: $(x_1, .., x_T)$

$$\mathrm{H}_{i+1} = FeedForward(A_{i+1}) \text{ and } A_{i+1} = SelfAttention(H_i) \quad \forall\, i \in [|1, L|]$$

$$\text{with} \quad SelfAttention(\mathrm{H}_i) = softmax(\frac{Q\,K^T}{\sqrt{\delta_K}})V$$

$$\mathrm{H}_0 = (Emb(x_1), ...Emb(x_T))$$

Transformer for Sequence Labeling

Given a sequence of tokens: $(x_1, .., x_T)$

$$\mathrm{H}_{i+1} = FeedForward(A_{i+1}) \text{ and } A_{i+1} = SelfAttention(H_i) \quad \forall i \in [|1, L|]$$

$$\text{with} \quad SelfAttention(\mathrm{H}_i) = softmax(\frac{Q\,K^T}{\sqrt{\delta_K}})V$$

$$\mathrm{H}_0 = (Emb(x_1), ...Emb(x_T))$$

- **Residual Connection** and **Layer Norm** are not included in those equations
- **FeedForward is position-wise** two layer MLP (i.e. applied independently from the position of each hidden vector)
- Self-Attention is actually a **Multi-Head Self-Attention**

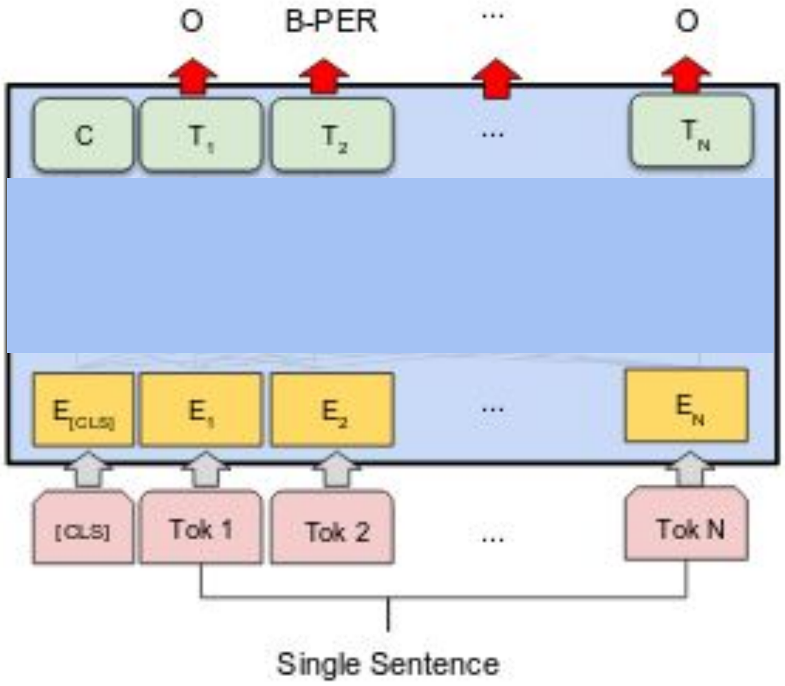
Transformer for Sequence Labeling

Given a sequence of tokens: $(x_1, .., x_T)$

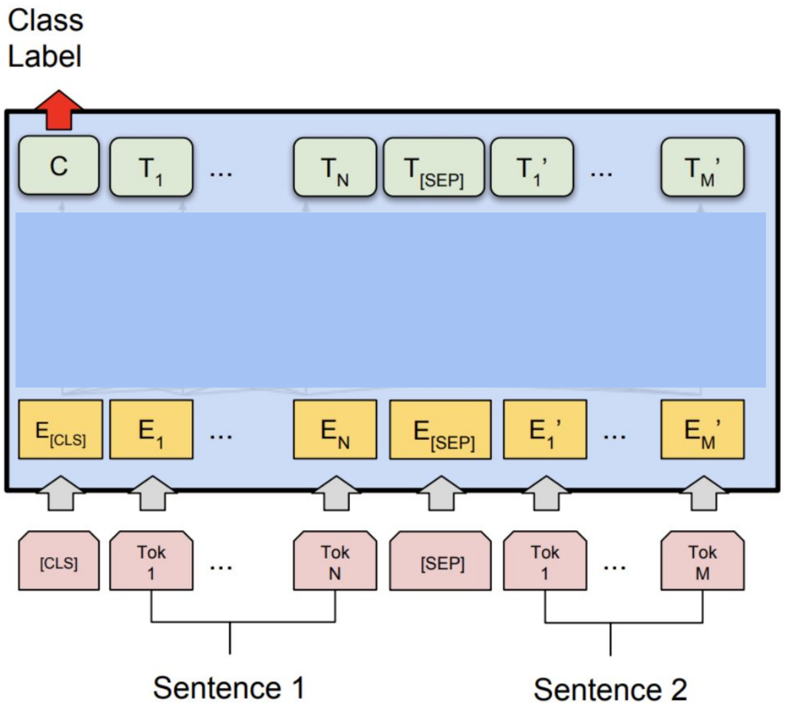
- **All the Hidden states of the last layer are fed to a softmax**

$$\boxed{\hat{p_{y_t}} = softmax(h_t) \ \ \forall t \leq \ T}$$

Transformer for Sequence Labeling



Transformer for Sequence Classification



Transformer for Sequence Labeling & Classification

Initialization:

- We can initialize randomly all the parameters of the model
- Train it on the sequence labeling & classification task with backpropagation

Still

- In practice, Transformer **underperforms LSTM models if we do that**

→ Not if we initialize our model in a **“smarter way“**

Pretraining with Mask-Language-Modeling

Pretraining with Mask-Language-Modeling

Let's take a **Transformer and Train it on a Language Modeling task**

We would like to have a Bidirectional Model

➔ **We introduce Mask Language Modeling**

Mask Language Modeling (MLM)

Given sequences of text, we *change* randomly 15% of tokens in each sequence

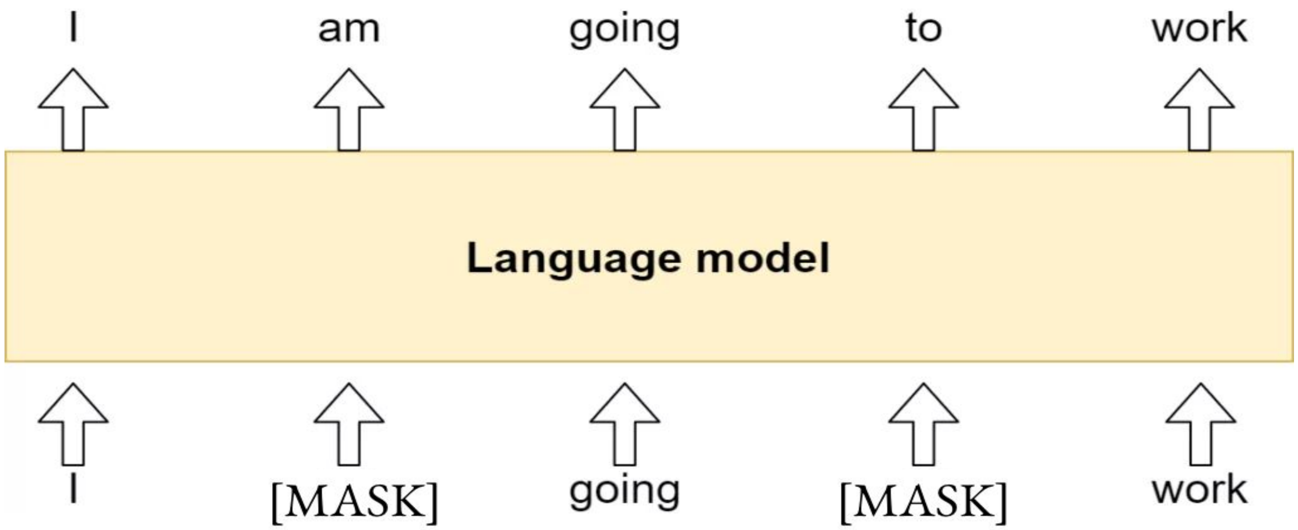
- **80%** of cases we replace them **with [MASK]**
- **20%** of cases we replace them **with a random token** of the vocabulary

MLM consists in **predicting** the changed tokens given the context (left and right)

Mask Language Modeling (MLM)

Given sequences of text, we *switch* randomly 15% of tokens in each sequence

- **80%** of cases we replace them **with [MASK]**
- **20%** of cases we replace them **with a random token** in the vocabulary



Transformer for MLM

- We train a large transformer: +12 layers
- On large dataset of raw text (+1GB up to 1TB) of text
- For many of steps: +100k steps

Transformer for MLM

- **We train a large transformer: +12 layers**
- **On large dataset of raw text (+1GB up to 1TB) of text**
- **For many of steps: +100k steps**

BERT, CamemBERT, Roberta, mBERT, XLM-R have been trained this way

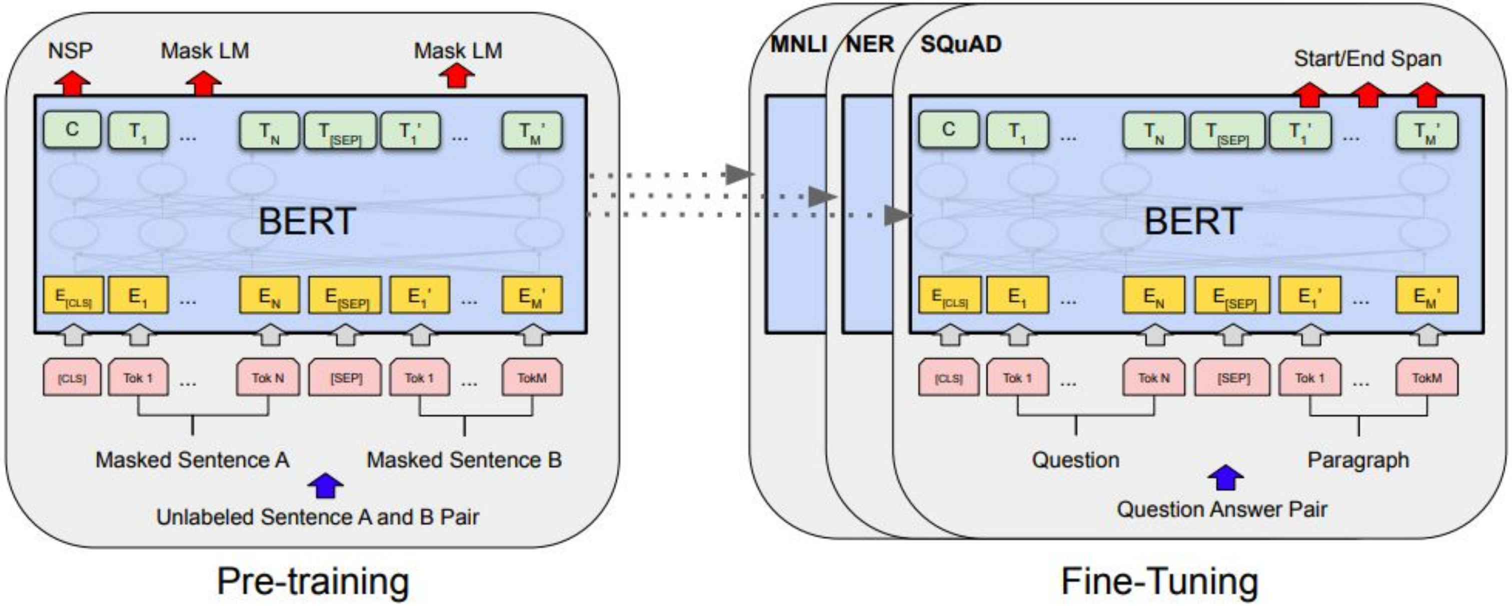
Transfer Learning with BERT-like Model

1. We ***pretrain*** a transformer model as described
2. We append a **task-specific Feed-Forward Layer on top**
3. We **fine-tune** the model **on the specific task** (labeling or classification)

   By fine-tuning, we simply mean keep training on the new labelled data after reusing all the parameters of the pretrained model

⇒ By doing this, we outperform LSTM models on ALL sequence labeling tasks

Transfer Learning with BERT-like Model



Transfer Learning with BERT-like Model

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Table: Performance of BERT vs. previous SOTA models on the GLUE benchmark (Devlin et. al 2018)

Transfer Learning with BERT-like Model

Intuition: Why does it work so well?

- **Language Modeling** is one of the **most challenging NLP task**
- **By reusing the pretrained model**, we re-use **very rich "representation" of the input sequences**
- **By fine-tuning the model on a specific task**, we adapt its parameters for the task

Hugging Face Hub

In a few lines of python code

- **Download**
- **Play with**
- **Fine-tune or Adapt**
- **Share**

+1000s pretrained Transformer models

Lecture Summary

- **POS Tagging, NER and BoolQA tasks**

- **Modeling those tasks**
  - **RNN/LSTM Models**
  - **Transformer**

- **Transfer Learning with Mask-Language-Modeling pretraining**