Fine-Tuning and Masked Language Models

Bidirectional Transformer Encoders

Bidirectional Transformer Encoder

We learned about causal language modeling

- Make autoregressive word prediction
- *Left-to-right transformers*

Now we introduce Bidirectional Transformer Encoder

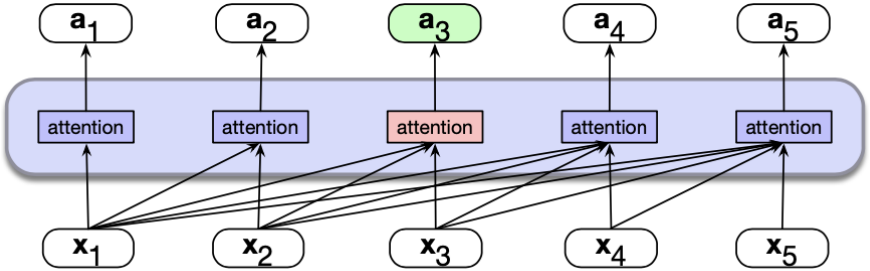
- Sees both left and right context
- Trained with *masked language modeling*

Bidirectional Transformer Encoder

BERT, RoBERTa, SpanBERT, XLM …

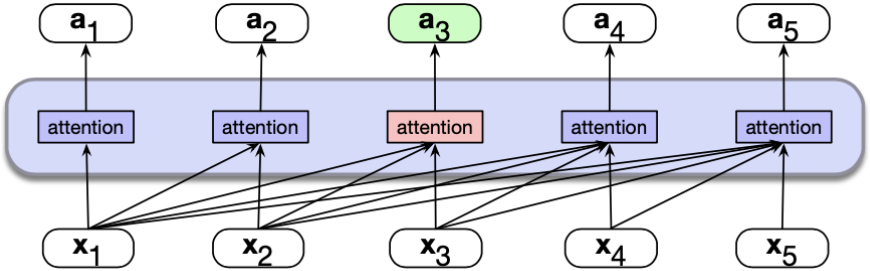
- They are used in downstream applications via *fine-tuning*
  - Pre-trained LM are used as input for a top layer network that is trained for the task
  - Pretraining/fine-tuning is a kind of *transfer learning*
- Embeddings generated by Masked Language models (MLM) are *contextual*

Bidirectional Transformer Encoder

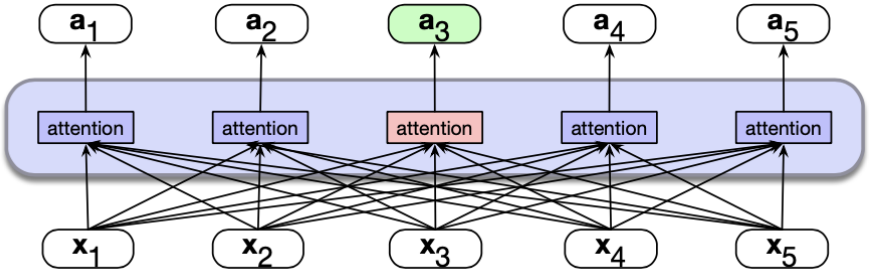


Left-to-right transformer architecture

Bidirectional Transformer Encoder



Left-to-right transformer architecture



Bidirectional transformer encoder architecture

Bidirectional Transformer Encoder

Classical multi-head self-attention mechanism

Single Head Attention

$$\mathbf{q}_i = \mathbf{W}^{Q}\mathbf{x}_i;\ \ \mathbf{k}_i = \mathbf{W}^{K}\mathbf{x}_i;\ \ \mathbf{v}_i = \mathbf{W}^{V}\mathbf{x}_i$$

$$\mathbf{y}_i = \sum_{j=1}^{n} \alpha_{ij}\mathbf{v}_j \qquad \alpha_{ij} = \frac{\exp(\text{score}_{ij})}{\sum_{k=1}^{n}\exp(\text{score}_{ik})}$$

$$\text{score}_{ij} = \mathbf{q}_i \cdot \mathbf{k}_j$$

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^{Q};\ \ \mathbf{K} = \mathbf{X}\mathbf{W}^{K};\ \ \mathbf{V} = \mathbf{X}\mathbf{W}^{V}$$

Multi-head Attention

Bidirectional Transformer Encoder

Parallel implementation of the Attention

N

q1·k1	−∞	−∞	−∞
q2·k1	q2·k2	−∞	−∞
q3·k1	q3·k2	q3·k3	−∞
q4·k1	q4·k2	q4·k3	q4·k4

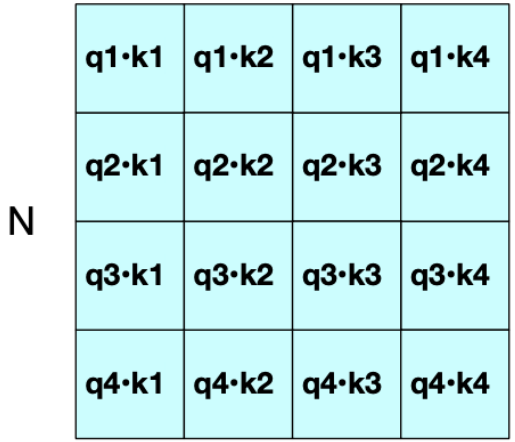
N

$$\mathbf{A} = \text{softmax}\left(\text{mask}\left(\frac{\mathbf{Q}\mathbf{K}^{\mathsf{T}}}{\sqrt{d_k}}\right)\right)\mathbf{V}$$

Bidirectional Transformer Encoder

Parallel implementation of the Attention

$$\mathbf{A} = \text{softmax}\left(\text{mask}\left(\frac{\mathbf{QK}^\intercal}{\sqrt{d_k}}\right)\right)\mathbf{V}$$



$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{QK}^\intercal}{\sqrt{d_k}}\right)\mathbf{V}$$

Bidirectional Encoder Representations from Transformers (BERT)

Subword tokenization

- WordPiece algorithm (variant of BPE)
- 30,000 subwords vocabulary
- 512 token+positional embeddings fixed input to control computational complexity
  - Memory requirement grows quadratically with the input size

Bidirectional Encoder Representations from Transformers (BERT)

WordPiece algorithm

- At each step, chooses the merge that most increases the language model probability of the tokenization.
- Uses a special character at the beginning of each token

words: Jet makers feud over seat width with big orders at stake
wordpieces: _J et _makers _fe ud _over _seat _width _with _big _orders _at _stake

Bidirectional Encoder Representations from Transformers (BERT)

WordPiece algorithm

1. Initialize the wordpiece lexicon with characters, collapsing all the remaining characters to a special unknown character token

2. Repeat until exactly the desired number *V* of token remain:

   a) Train a n-gram language model on the training corpus, using the current set of wordpieces

   b) Consider the set of possible new wordpieces made by concatenating two wordpieces from the current lexicon. Choose the one that most increases the language model probability of the training corpus.

Bidirectional Encoder Representations from Transformers (BERT)

Multi-layer transformer blocks

- 768 units in heach hidden layer
- $\text{BERT}_{\text{BASE}}$: 12 transformer layers, 12 self-attention heads per layer, 110M params
- $\text{BERT}_{\text{LARGE}}$: 24 transformer layers, 16 self-attention heads per layer, 304M params

Fine-Tuning and Masked Language Models

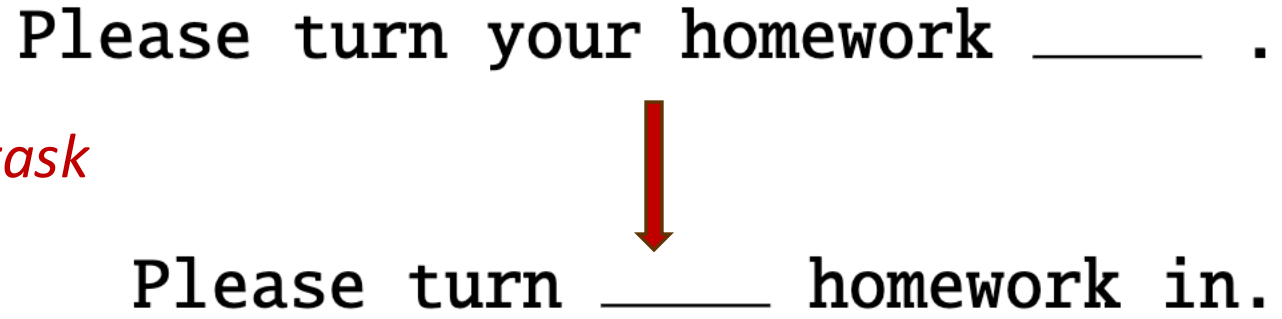
Training Bidirectional Encoders

Masked Language Modeling

Causal language modeling becomes trivial when we remove the causal mask

Masked Language Modeling uses a «fill-in-the-blank» approach

- The *cloze task*



Masked Language Modeling

During training the model is deprived of one or more elements of an input sequence

It must generate a probability distribution over the vocabulary for each of the missing items

We then use the cross-entropy loss from each of the model's predictions to drive the learning process

Masked Language Modeling

Masking can be generalized to any method that corrupt the training input and then asks the model to recover the original input.

- Masks
- Substitutions
- Reorderings
- Deletions
- Extraneous insertions into the training text

Masking words

BERT makes use of unannotated large text corpora

A random sample of tokens from each training sequence is selected for use in the learning task

Masking words

Once selected, a token is used in one of the following three ways:

- It is replaced with the unique vocabulary token `[MASK]`
- It is replaced with another token from the vocabulary, randomly sampled based on token unigram probabilities
- It is left unchanged

Masking words

In BERT 15% tokens are sampled for learning

- 80% is substitued by the `[MASK]` token
- 10% is replaced with a random token
- 10% is left unchanged

All the tokens are involved in self-attention, but only the selected ones are used in computing the loss through the softmax

Masking words

$$L_{MLM} = -\frac{1}{|M|}\sum_{i \in M} \log P(x_i|\mathbf{h}_i^L)$$

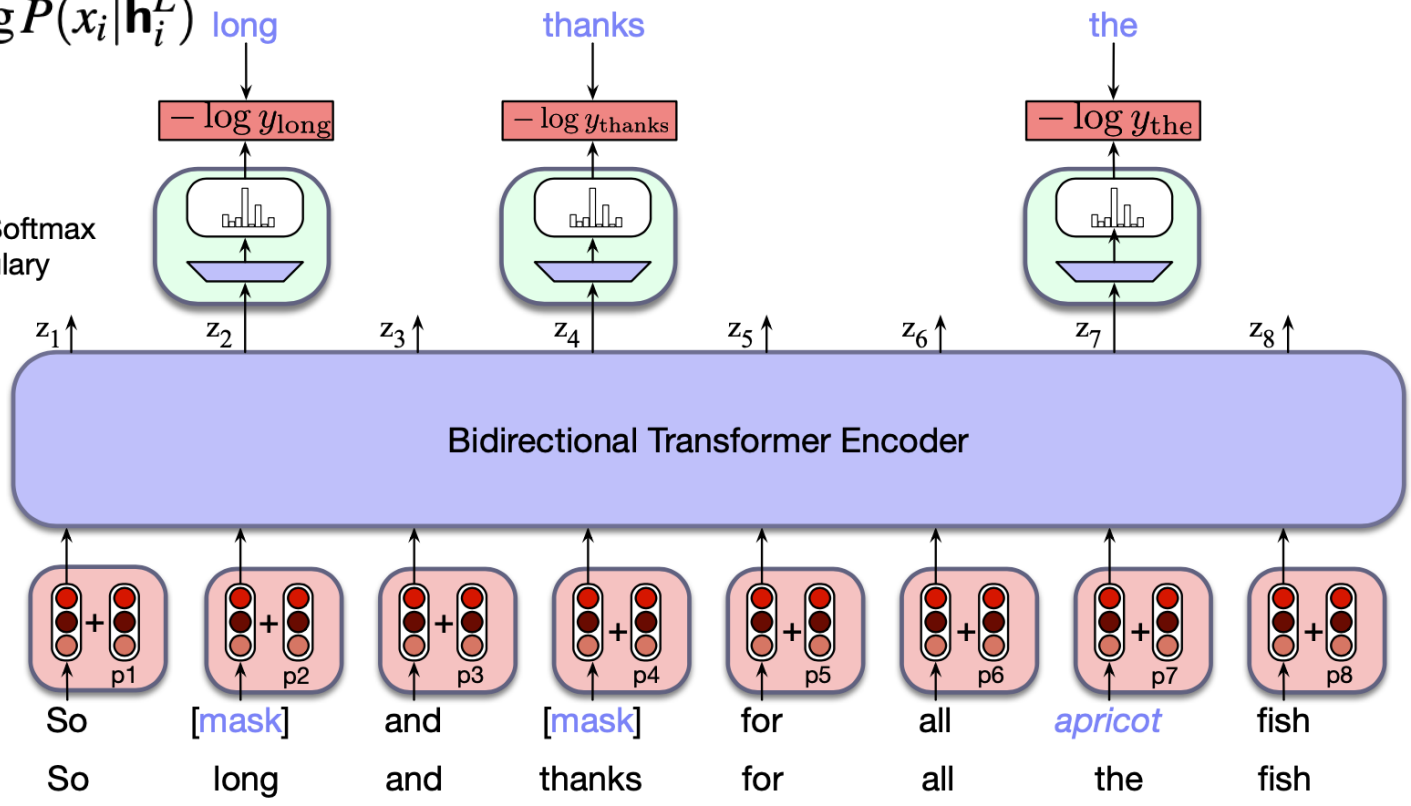
$$L_{MLM}(x_i) = -\log P(x_i|\mathbf{h}_i^L)$$

CE Loss

LM Head with Softmax over Vocabulary

$$\mathbf{u}_i = \mathbf{h}_i^L \mathbf{E}^T$$

$$\mathbf{y}_i = \text{softmax}(\mathbf{u}_i)$$



Token + Positional Embeddings

Next sentence prediction

Many NLP tasks involve determining the relation between pairs of sentences

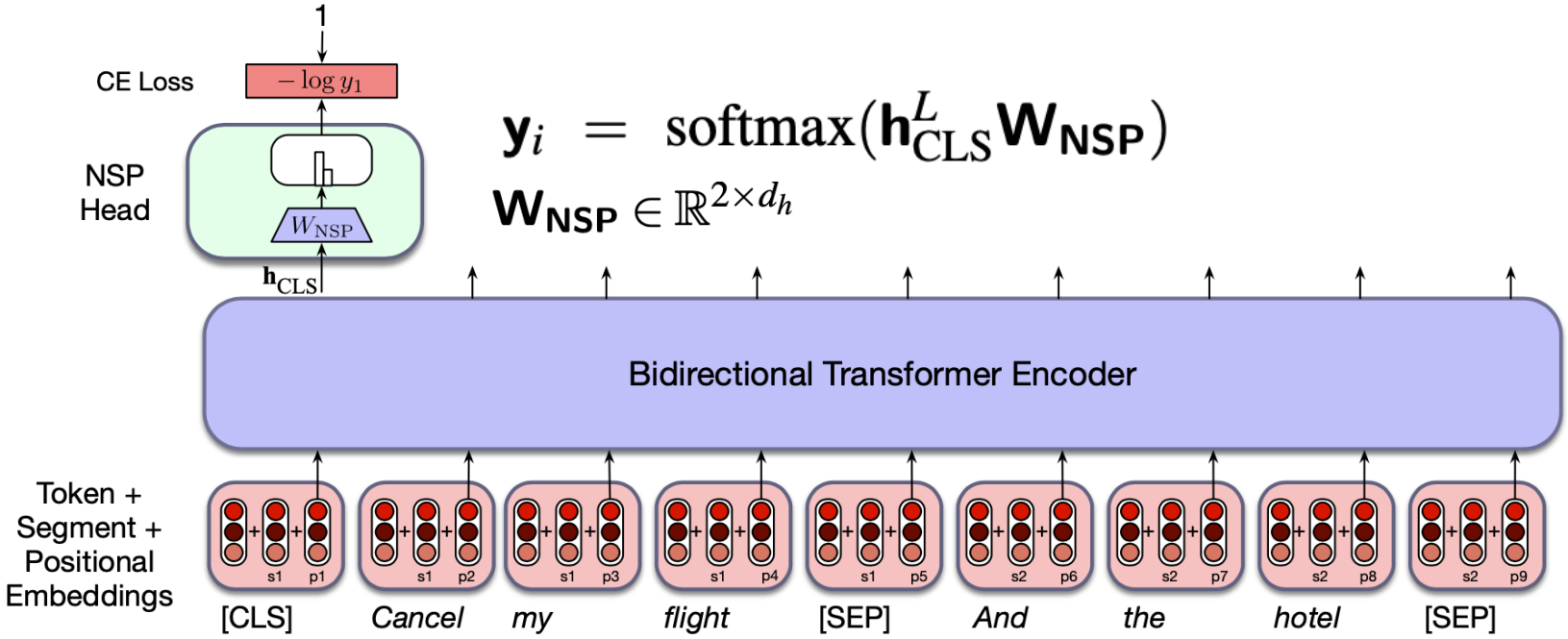
- Paraphrase detection
- Textual entailment
- Discourse coherence
- …

Next sentence prediction

BERT achieves Next Sentence Prediction (NSP) using the following training procedures

- 50% split between positive (related) and negative(unrelated) sentence pairs
- Two new tokens are introduced
  - `[CLS]` is prepended to the input sentence pair
  - `[SEP]` is placed between the sentences and after the final token

Next sentence prediction



BERT training

Trained on

- BookCorpus (800M words)
- English Wikipedia (2500M words)

NSP using the 50%/50% scheme

- Sentence pairs sampled to fit in the 512 token input size

40 epochs training using MLM plus NSP loss

Fine-Tuning and Masked Language Models

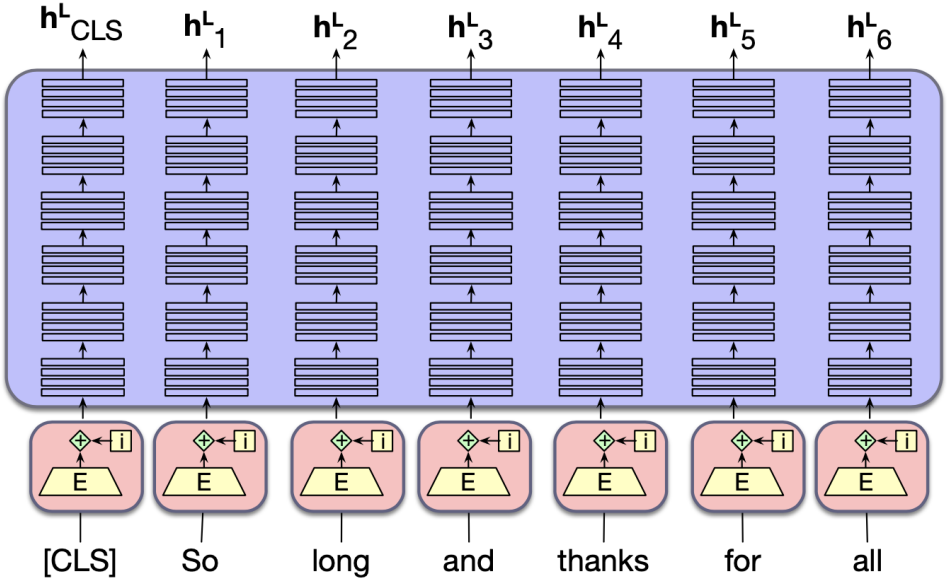
Contextual embeddings

Contextual embeddings

The output provided by the final layer of a pretrained LM for each token $\boldsymbol{x}_i$ of a novel sentence can be regarded as a *contextual embedding*

- Vectors representing some aspect of the meaning of a *word instance* in its context
- Often, the c.e. is obtained by averaging the output of the last four layers of the model

Contextual embeddings



- Static embeddings → meaning of *word types*
- Contextual embeddings → meaning of *word instances*

ContextualEmbedding(«long») → $\mathbf{h}^L_2$

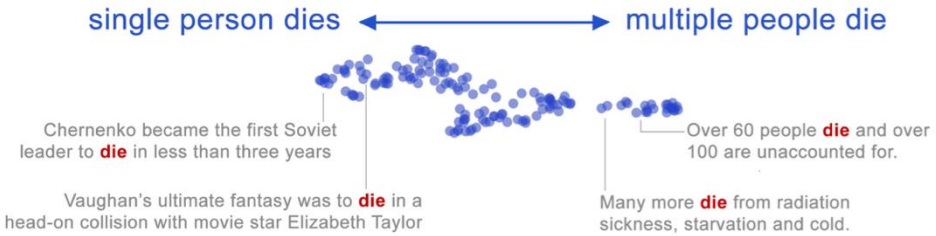
Contextual Embeddings and Word Sense

Words are *ambiguous*

- Many meanings for the same wordform → Polisemy
- Remember the synsets in WordNet

Contextual Embeddings and Word Sense

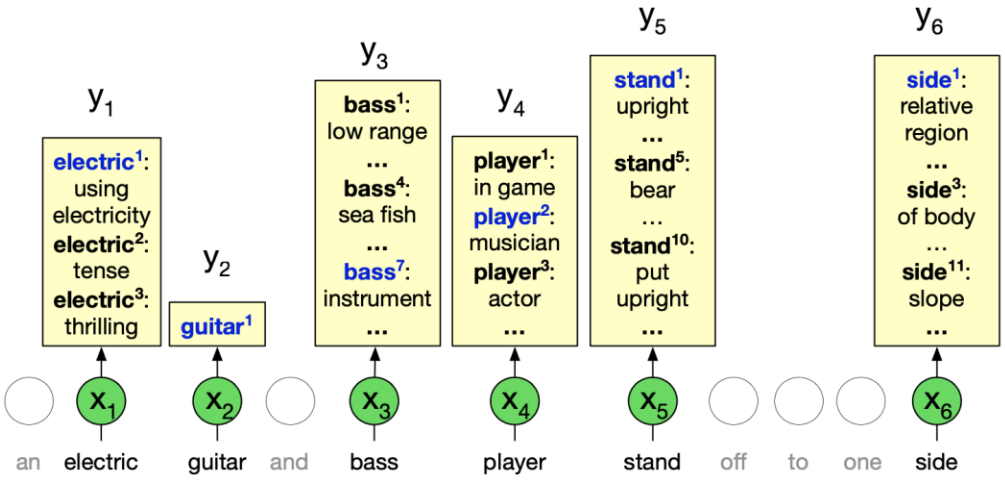
Words are *ambiguous*



Contextual Embeddings and Word Sense

Word sense disambiguation (WSD)

- The task of selecting the correct sense for a word
- WSD algorithms take as input a word in context and the WordNet synsets; they output the correct word sense in context.



Contextual Embeddings and Word Sense

1-nearest neighbour WSD

1. For each token *i* in a sense-labeled data set (i.e. SenseEval)
   a) Compute the contextual embedding $\mathbf{v}_i$
2. For each sense *s* made by *n* tokens
   a) Compute the *sense embedding*

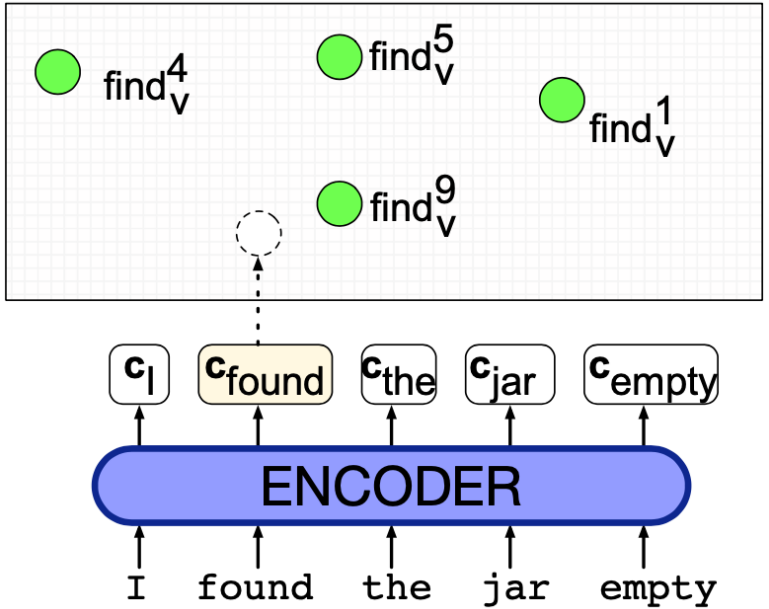
$$\mathbf{v}_s = \frac{1}{n}\sum_i \mathbf{v}_i \qquad \forall \mathbf{v}_i \in \text{tokens}(s)$$

Contextual Embeddings and Word Sense

1-nearest neighbour WSD

1. Given a target word *t* in context compute the contextual embedding **t**
2. Provide the sense as

$$\text{sense}(t) = \underset{s \in \text{senses}(t)}{\text{argmax}} \ \text{cosine}(\mathbf{t}, \mathbf{v}_s)$$



Contextual Embeddings and Word Similarity

Contextual embeddings provided by either masked or autoregressive LMs are *anisotropic*

- They tend more or less in the same direction so their cosine distance roughly equals 1
- Very few dimensions in a contextual embedding dominate the others

Contextual Embeddings and Word Similarity

Contextual embeddings provided by either masked or autoregressive LMs are *anisotropic*

- One can try to standardize contextual embedding using z-scoring but the cosine tends to underestimate human judgements on WSD for very frequent words
- In general they are used as inputs for fine-tuned classifiers in downstream NLP applications

Fine-Tuning and Masked Language Models

Transfer Learning through Fine-Tuning

Fine-tuning

Pretrained LMs are powerful at generalizing

- Useful for downstream applications

Two strategies for fine-tuning

- *Prompt* the model, while putting it in a state where it contextually generates what we want
- *Add application-specific circuitry* (a *head*) on top of pretrained LMs, taking their output as its input

Fine-tuning

When using an application-specific head

- Train the new parameters using labeled application data
- Freeze or make slight adjustments to the LM parameters

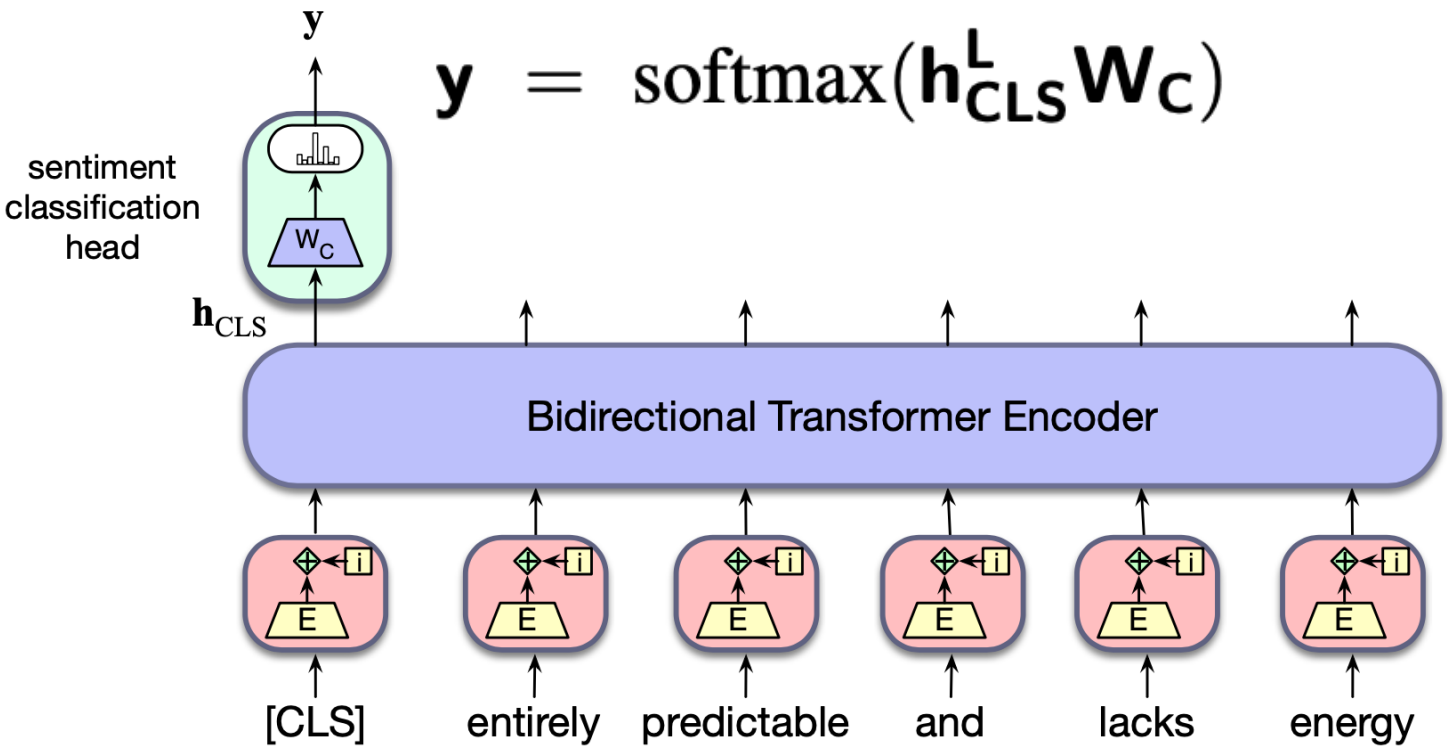
Sequence Classification

In sequence classification (i.e. sentiment analysis) a *sentence embedding* is often used to allow classification

In BERT the `[CLS]` token plays the role of sentence embedding that is the input of a *classifier head*

A dense layer $\mathbf{W}_c$ is fine-tuned with labeled data and cross-entropy loss

Sequence Classification



Pair-Wise Sequence Classification

Recognizing the relation between pairs of sentences: textual entailment, dicourse coherence …

Fine-tuning proceeds the same as NSP training

- Sentence pairs are presented to the model (`[SEP]` is used)
- The `[CLS]` token head is trained with class labels

Pair-Wise Sequence Classification

Example: textual entailment with the MultiNLI corpus

- `Neutral`
  - a: Jon walked back to the town to the smithy.
  - b: Jon traveled back to his hometown.
- `Contradicts`
  - a: Tourist Information offices can be very helpful.
  - b: Tourist Information offices are never of any help.
- `Entails`
  - a: I'm confused.
  - b: Not all of it is very clear to me.

Sequence Labelling

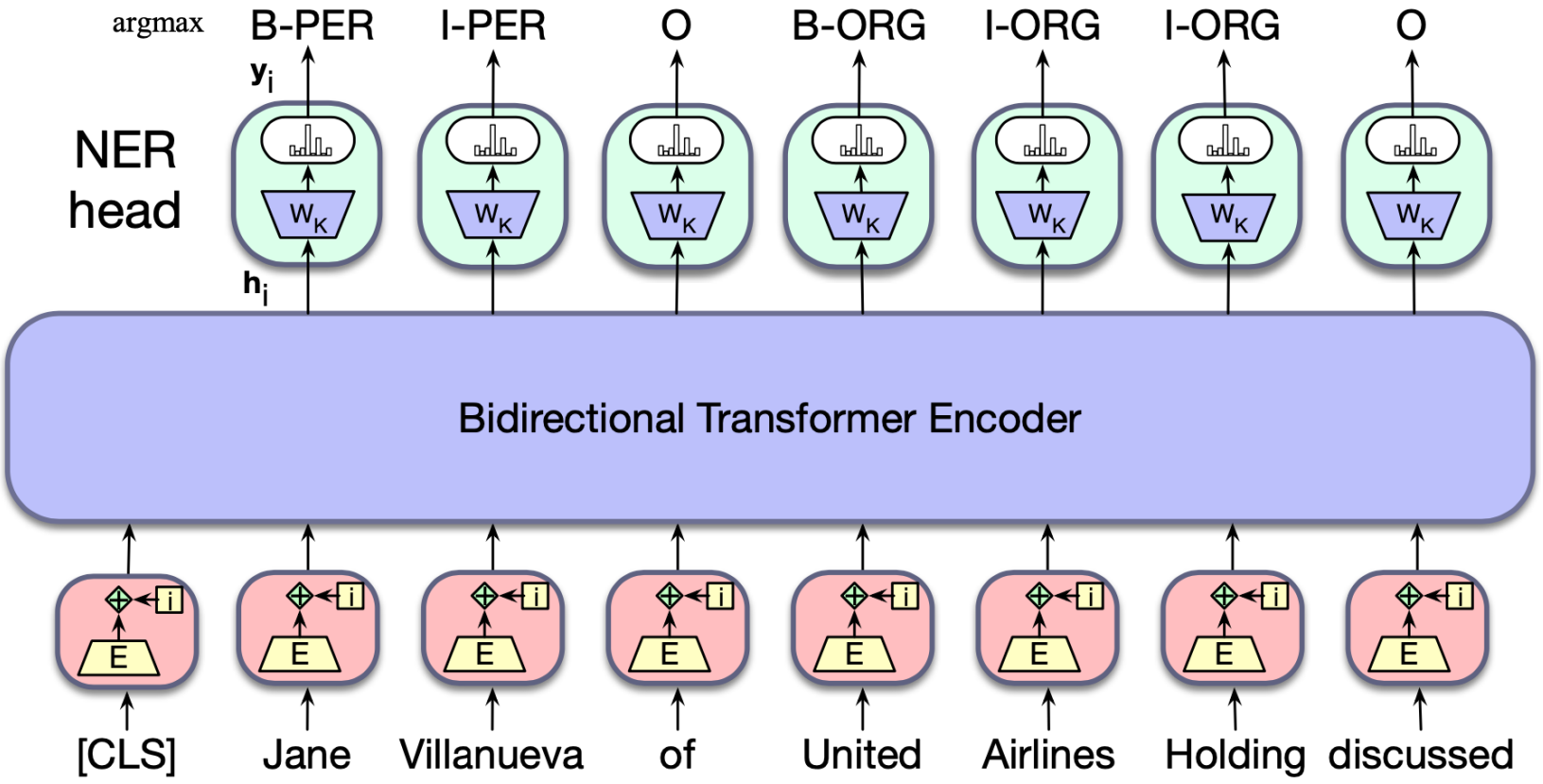
Example: BIO-based Named Entity Recognition

A dense layer $\mathbf{W_K}$ is trained on the output provided separately for each input token

- $\mathbf{W_K}$ has *k* columns, one for each tag
- The *argmax* is used to select the output tag

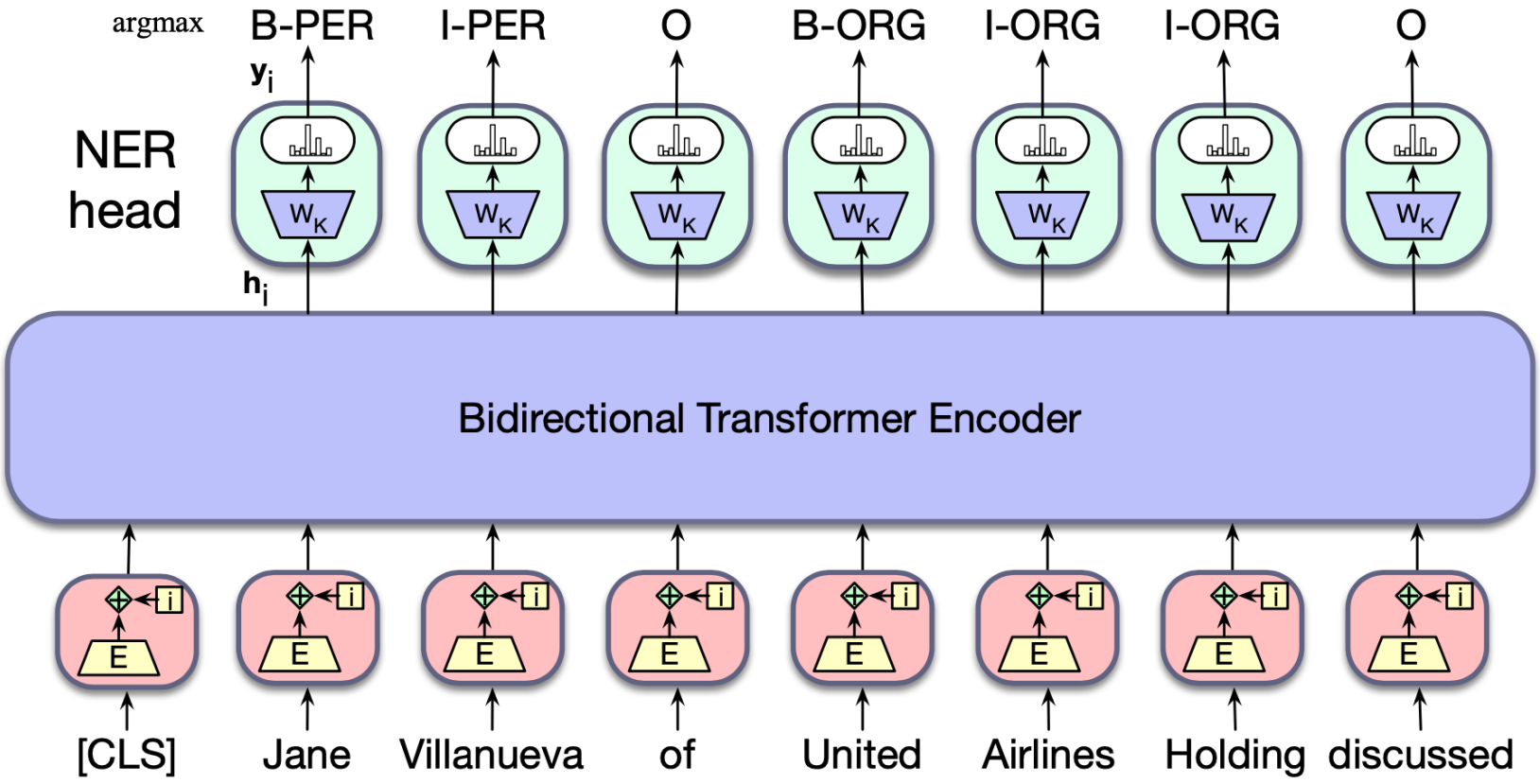
Sequence Labelling

$$\mathbf{y_i} = \text{softmax}(\mathbf{h_i^L W_K})$$
$$\mathbf{t_i} = \text{argmax}_k(\mathbf{y}_i)$$



Sequence Labelling

A CRF layer can be used over the y_i values to take global tag-level transitions into account



Sequence Labelling

A problem: WordPiece sub-word tokenization

Example:

[LOC **Mt. Sanitas**] is in [LOC **Sunshine Canyon**] .

Mt. Sanitas is in Sunshine Canyon .

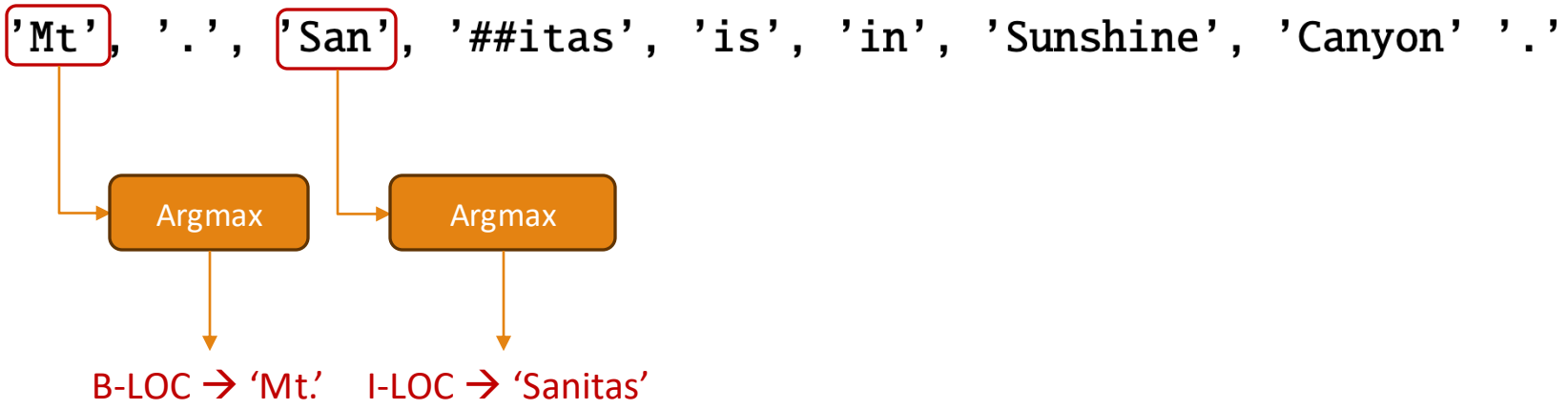
B-LOC I-LOC O O B-LOC I-LOC O

`'Mt', '.', 'San', '##itas', 'is', 'in', 'Sunshine', 'Canyon' '.'`

B-LOC B-LOC I-LOC I-LOC O O B-LOC I-LOC O

Sequence Labelling

Decoding through *argmax* of the tag assigned to the first sub-word token



Fine-Tuning and Masked Language Models

BERT variants

Multilingual BERT

Trained on 104 languages at the same time to build a shared representation of the word meaning in each of them

110K token shared vocabulary for WordPiece

Not the best in cross-lingual tasks

RoBERTa

Developed by Facebook (now Meta)

Trained on five English corpora (160GB uncompressed text)

Effective pre-training with *dynamic masking*

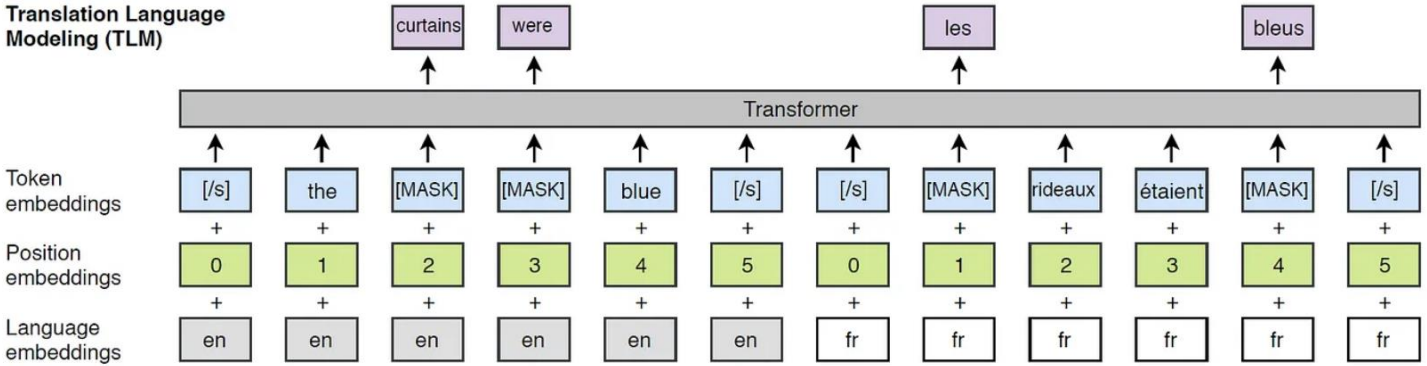
- Masking changes at each input sequence
- BERT makes the masking just once as preprocessing

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

XLM (croX lingual Language Model)

Developed by Facebook (pre-training on 64 GPUs)

Translation Language Modelling (TLM)



Lample, G., & Conneau, A. (2019). Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

XLM-RoBERTa

Developed by Facebook

Much more computational power (pre-training on 500 GPUs) and much more data

It doesn’t use language embeddings to enable a fast language switch

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., … & Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.