UPPSALA UNIVERSITET

Tranformers and BERT

Måns Magnusson
Statistiska Institutionen
Uppsala Universitet

17th of June 2024

UPPSALA UNIVERSITET

Section 1

Introduction

The Development of NLP

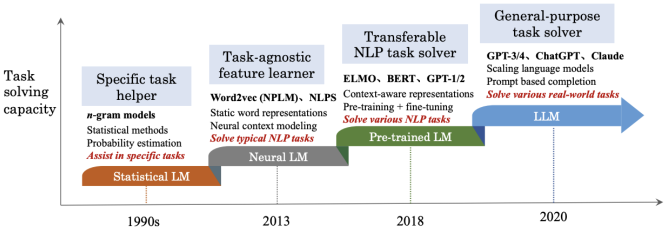

Figure: The development timeline (Zhao et al., 2023)

UPPSALA UNIVERSITET

Why and when neural networks?

Why and when neural networks?

- Learning feature representations

UPPSALA
UNIVERSITET

Why and when neural networks?

- Learning feature representations
- Good for "sensor" data

Why and when neural networks?

- Learning feature representations
- Good for "sensor" data
- Needs a lot of data to learn complex representations (image, text, audio)

Learning Representations

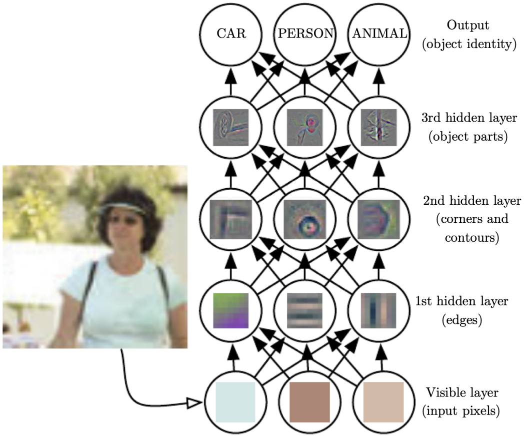

Figure: Learning representations can be crucial (Goodfellow et al, 2017, Fig. 1.2)

Different Network Architectures

- Different networks for different purposes
  - Feed-Forward Neural Network: Basic building block

Different Network Architectures

- Different networks for different purposes
  - Feed-Forward Neural Network: Basic building block
  - Convolutional Neural Networks: Computer Vision

Different Network Architectures

- Different networks for different purposes
  - Feed-Forward Neural Network: Basic building block
  - Convolutional Neural Networks: Computer Vision
  - Transformers: Textual data

Section 2

(Recap) Feed-Forward Neural Networks

The Feed-Forward Network

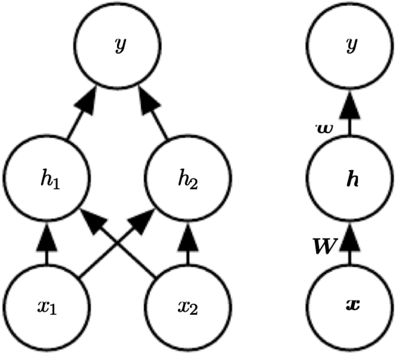

Figure: A simple feed-forward network (Goodfellow et al, 2017, Fig. 6.2)

Important concepts:

Layers, neurons, input, output, weights, bias, architecture

The Feed-Forward Network

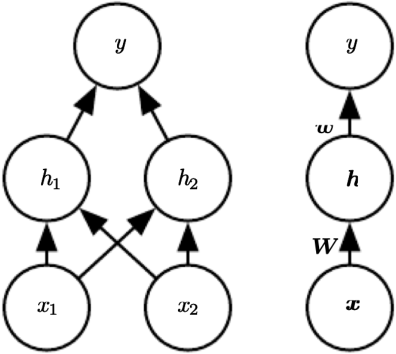

Figure: A simple feed-forward network (Goodfellow et al, 2017, Fig. 6.2)

In mathematical notation:

$$y_i = \mathbf{w}^T g(\mathbf{W}^T \mathbf{x}_i + \mathbf{b}_1) + \mathbf{b}_2 \,,$$

where $\mathbf{w}$, $\mathbf{W}$, $\mathbf{b}_1$ and $\mathbf{b}_2$ is learned/estimated.

The Feed-Forward Network

$$y_i = \mathbf{w}^T g(\mathbf{W}^T \mathbf{x}_i + \mathbf{b}_1) + \mathbf{b}_2$$

UPPSALA UNIVERSITET

The Feed-Forward Network

$$y_i = \mathbf{w}^T g(\mathbf{W}^T \mathbf{x}_i + \mathbf{b}_1) + \mathbf{b}_2$$

$$W = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, w = \begin{pmatrix} 1 \\ -2 \end{pmatrix}, b_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, b_2 = \begin{pmatrix} 0 \end{pmatrix}$$

The Feed-Forward Network

$$y_i = \mathbf{w}^T g(\mathbf{W}^T \mathbf{x}_i + \mathbf{b}_1) + \mathbf{b}_2$$

$$W = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, w = \begin{pmatrix} 1 \\ -2 \end{pmatrix}, b_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, b_2 = \begin{pmatrix} 0 \end{pmatrix}$$

$$g(z) = \mathrm{ReLU}(z) = \max(0, z)$$

UPPSALA UNIVERSITET

The Feed-Forward Network

$$y_i = \mathbf{w}^T g(\mathbf{W}^T \mathbf{x}_i + \mathbf{b}_1) + \mathbf{b}_2$$

$$W = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, w = \begin{pmatrix} 1 \\ -2 \end{pmatrix}, b_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, b_2 = (0)$$

$$g(z) = \mathrm{ReLU}(z) = \max(0, z)$$

$$x_i = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

The Feed-Forward Network

$$y_i = \mathbf{w}^T g(\mathbf{W}^T \mathbf{x}_i + \mathbf{b}_1) + \mathbf{b}_2$$

$$W = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, w = \begin{pmatrix} 1 \\ -2 \end{pmatrix}, b_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, b_2 = \begin{pmatrix} 0 \end{pmatrix}$$

$$g(z) = \mathrm{ReLU}(z) = \max(0, z)$$

$$x_i = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

$$y_i = \begin{pmatrix} 1 \\ -2 \end{pmatrix}^T \mathrm{ReLU}\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ -1 \end{pmatrix}\right] + \begin{pmatrix} 0 \end{pmatrix}$$

The Feed-Forward Network

$$y_i = \mathbf{w}^T g(\mathbf{W}^T \mathbf{x}_i + \mathbf{b}_1) + \mathbf{b}_2$$

$$W = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, w = \begin{pmatrix} 1 \\ -2 \end{pmatrix}, b_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, b_2 = (0)$$

$$g(z) = \text{ReLU}(z) = \max(0, z)$$

$$x_i = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

$$y_i = \begin{pmatrix} 1 \\ -2 \end{pmatrix}^T \text{ReLU}\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ -1 \end{pmatrix}\right] + (0)$$

$$y_i = \begin{pmatrix} 1 \\ -2 \end{pmatrix}^T \begin{pmatrix} 1 \\ 0 \end{pmatrix} + (0) = 1$$

UPPSALA UNIVERSITET

Output units (g_L)

- Depend on the data y

Output units (g_L)

- Depend on the data y
- Linear units for regression

$$\hat{y} = \mathbf{wh} + \mathbf{b}$$

Output units (g_L)

- Depend on the data y
- Linear units for regression

$$\hat{y} = \mathbf{wh} + \mathbf{b}$$

- Bernoulli units for binary classification

$$\hat{y} = \sigma(\mathbf{wh} + \mathbf{b}),$$

where

$$\sigma(z) = \frac{1}{1 + \exp(-z)},$$

i.e. the logistic or sigmoid function.

- Other likelihoods can be used, such as Multinomial, Poisson, etc.

Activation functions (g_l)

- Historically $g(z)$ has been the sigmoid or or hyperbolic tangent (tanh)

$$g_{\text{sigmoid}}(z) = \frac{e^z}{e^z + 1} = \frac{1}{1 + e^{-z}}$$

$$g_{\text{tanh}}(z) = \frac{\sinh z}{\cosh z} = \frac{e^{2z} - 1}{e^{2z} + 1}$$

Activation functions (g_l)

- Historically $g(z)$ has been the sigmoid or or hyperbolic tangent (tanh)

$$g_{\text{sigmoid}}(z) = \frac{e^z}{e^z + 1} = \frac{1}{1 + e^{-z}}$$

$$g_{\tanh}(z) = \frac{\sinh z}{\cosh z} = \frac{e^{2z} - 1}{e^{2z} + 1}$$

- Today, variants of Rectified linear unit (ReLU) is common
  - Easier to estimate with SGD
  - Easier for deep models

Activation functions (g_l): ReLU

$$g_{\text{ReLU}}(z) = \max(0, z)$$

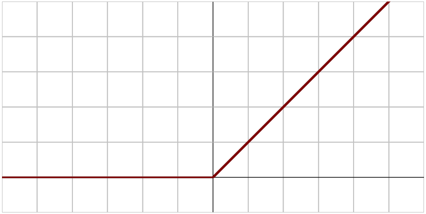

Figure: Rectified Linear Unit (Wikipedia)

UPPSALA UNIVERSITET

Activation functions (g_l): GeLU

UPPSALA UNIVERSITET

$$g_{\text{GeLU}}(z) = z\Phi(z)$$

where $\Phi(z)$ is a standard Gaussian.

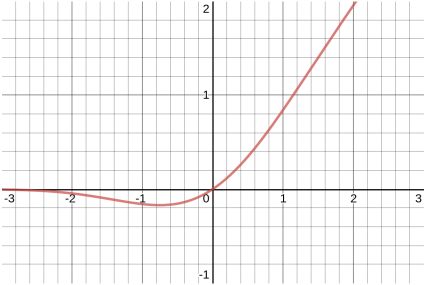

Figure: Gaussian Error Linear Unit (Wikipedia)

Activation functions (g_l): ELU

$$g_{\mathrm{ELU}}(z) = \begin{cases} \alpha\left(e^z - 1\right) & \text{if } z \leq 0 \\ z & \text{if } z > 0 \end{cases}$$

where α is commonly 1.

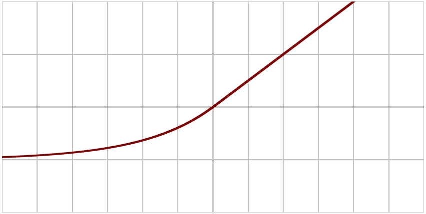

Figure: Exponential Linear Unit (Wikipedia)

Architecture design

- Architecture: the overall structure of the network
- Choices:
  - How many layers?
  - How many hidden units in each layer?
  - Activation functions?

UPPSALA UNIVERSITET

Depth matters

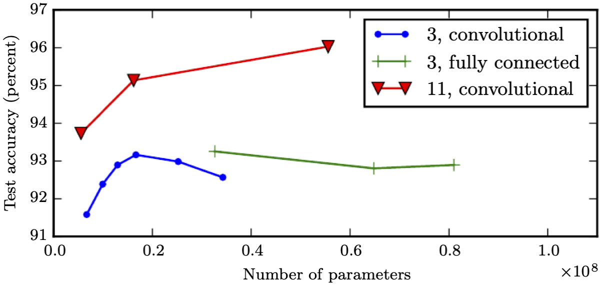

Figure: Depth vs. no of parameters (Goodfellow et al, 2017, Fig. 6.3)

Depth matters II

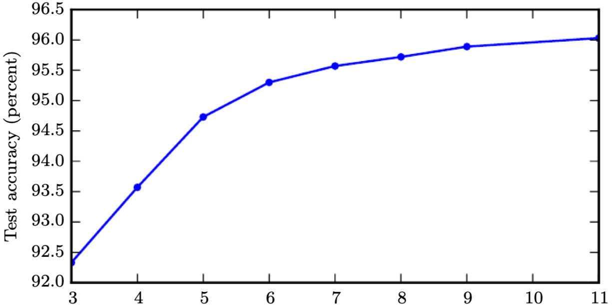

Figure: Effect of depth on accuracy (Goodfellow et al, 2017, Fig. 6.6)

Section 3

The Transformer

The Transformer

- Introduced by Vaswani et al. (2017):
  Attention is all you need.

The Transformer

- Introduced by Vaswani et al. (2017):
  Attention is all you need.
- Behind the recent progress in NLP: BERT, Llama, GPT, etc.

The Transformer

- Introduced by Vaswani et al. (2017):
  Attention is all you need.
- Behind the recent progress in NLP: BERT, Llama, GPT, etc.
- Benefits for textual data:
  - Enables more GPU parallelism
  - Better handling of long-range dependencies
  - Enable transfer learning for text data
  - Enables deeper networks for text

UPPSALA UNIVERSITET

The Transformer

- Introduced by Vaswani et al. (2017): Attention is all you need.
- Behind the recent progress in NLP: BERT, Llama, GPT, etc.
- Benefits for textual data:
  - Enables more GPU parallelism
  - Better handling of long-range dependencies
  - Enable transfer learning for text data
  - Enables deeper networks for text
- I will rely heavily on images from Allamar (2018) The Illustrated Transformer (recommended)

A Sequence-to-Sequence Model

Figure: The basic block (Allamar, 2018)

Stacked Encoder-Decoder Structure

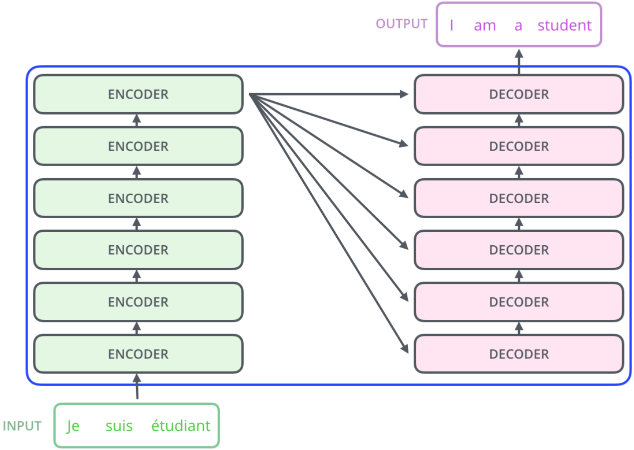

Figure: The Transformer layers (Allamar, 2018)

Transformer

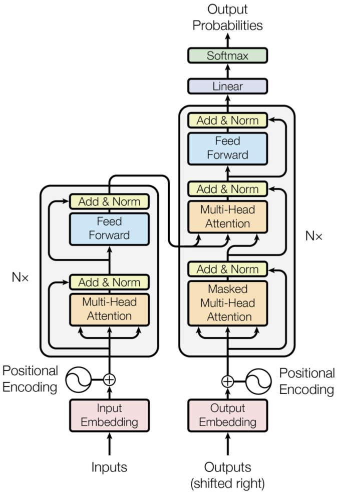

Figure: The Transformer Architecture (Vaswani et al., 2017)

The encoder vs. the decoder

- Encoder:
  - Input: words
  - Output: contextualized embeddings
- Decoder:
  - Input: previous words (and contextualized embeddings from encoder)
  - Output: next word prediction

The Transformer Layer (Encoder layer)

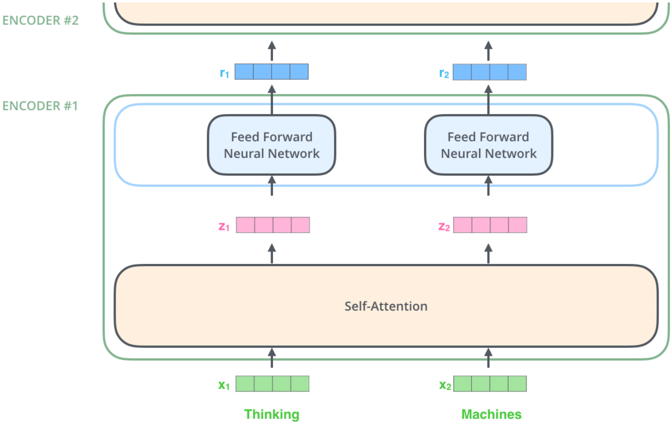

Figure: The Encoder Layer (Alammar, 2018b)

UPPSALA UNIVERSITET

Scaled Dot-Product Attention

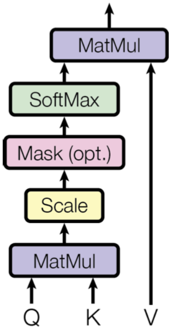

Figure: Scaled Dot-Product Attention (Vaswani et al., 2017)

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{QK}^T}{\sqrt{d_k}}\right)\mathbf{V}$$

Attention components

- (Q)uery: Word i query other words
- (K)ey: The other words return their key to i
- (V)alue: The value of the other words to i

Computing Q, V and K

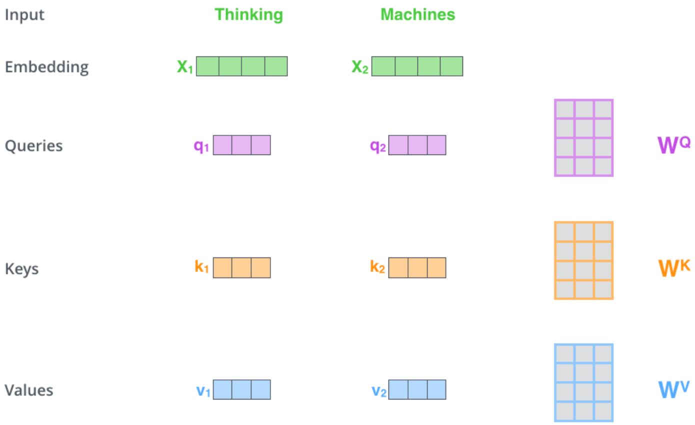

Figure: Attention heads (Alammar, 2018b)

Computing Self-Attention

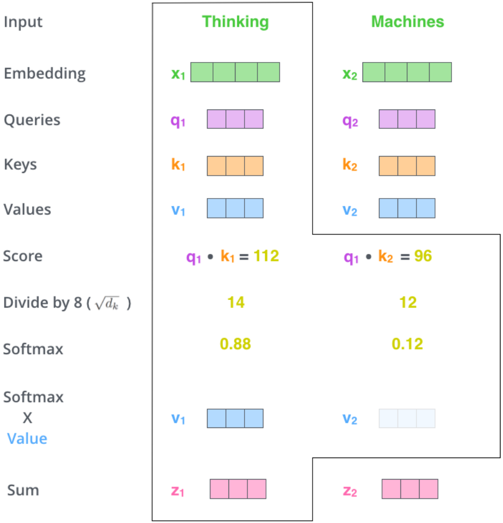

Figure: Attention (Alammar, 2018b)

Multi-Head Attention

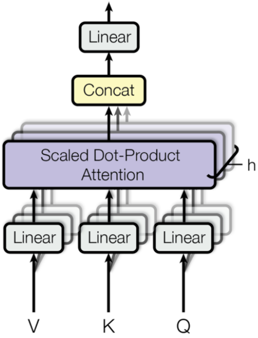

Figure: Scaled Dot-Product Attention (Vaswani et al., 2017)

Attentions Heads

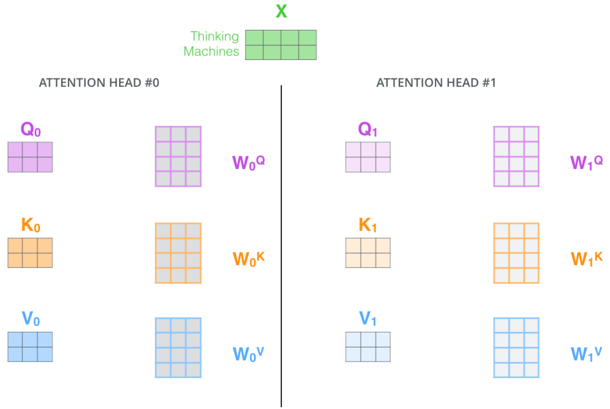

Figure: Attention heads (Alammar, 2018b)

Multi-head attention

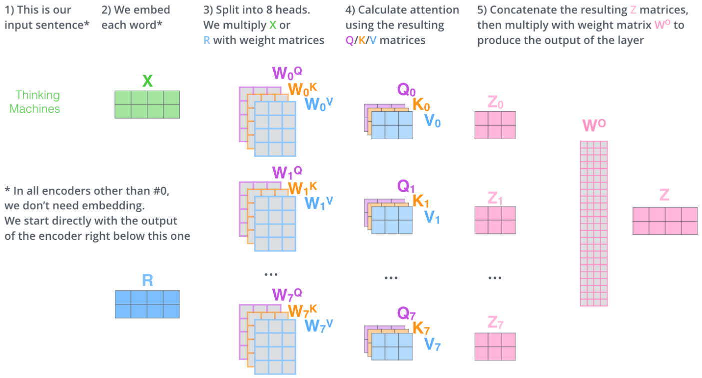

Figure: Attention heads (Alammar, 2018b)

UPPSALA UNIVERSITET

- Introduction
- (Recap) Feed-Forward Neural Networks
  - Hidden Units
  - Architecture design
- The Transformer
  - Attention
  - Multi-Head Attention
  - Positional encoding
  - Add and Normalize
- Transformer-Encoder Models
  - BERT
  - RoBERTa

Multi-Head Attention example

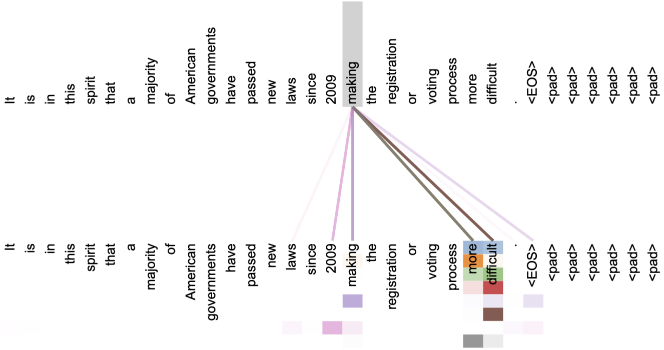

Figure 3: An example of the attention mechanism following long-distance dependencies in the encoder self-attention in layer 5 of 6. Many of the attention heads attend to a distant dependency of the verb 'making', completing the phrase 'making...more difficult'. Attentions here shown only for the word 'making'. Different colors represent different heads. Best viewed in color.

Figure: Attention (Vaswani et al., 2017)

UPPSALA UNIVERSITET

Positional Encoding

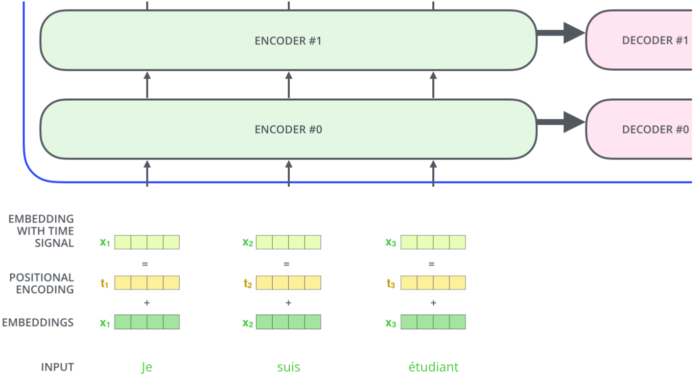

Figure: Attention heads (Alammar, 2018b)

(Absolute) Positional Encoding

"The boy hit the ball" vs "The ball hit the boy"

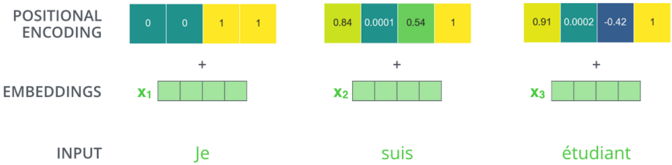

Figure: Adding positional encodings to embeddings (Alammar, 2018b)

Positional encodings

"The boy hit the ball" vs "The ball hit the boy"

1. Absolute position encoding
2. Relative position encoding:
   The distance between tokens are added as bias terms

Positional encodings

"The boy hit the ball" vs "The ball hit the boy"

1. Absolute position encoding
2. Relative position encoding:
   The distance between tokens are added as bias terms
3. Rotational positional encoding (RoPE, Su et al., 2022):
   Q and K are rotated by based on distance

UPPSALA UNIVERSITET

Add and Normalize

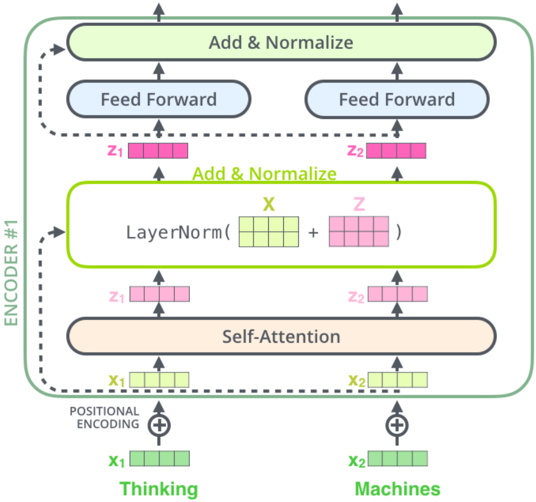

Figure: Add and Normalize (Alammar, 2018b)

UPPSALA UNIVERSITET

Transformer

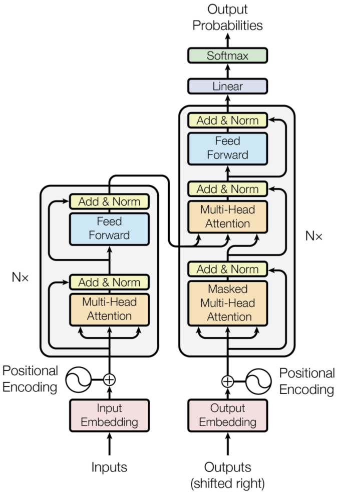

Figure: The Transformer Architecture (Vaswani et al., 2017)

Tokenization

- Subword tokenization is commonly used

UPPSALA UNIVERSITET

Tokenization

- Subword tokenization is commonly used
- The main problem with tokenization
  1. Very large vocabulary size
  2. Out-of-vocabulary (OOV) tokens
  3. Different meanings of very similar words

UPPSALA UNIVERSITET

Tokenization

- Subword tokenization is commonly used
- The main problem with tokenization
  1. Very large vocabulary size
  2. Out-of-vocabulary (OOV) tokens
  3. Different meanings of very similar words
- Two common approaches:
  1. Byte-pair encoding (GPT-2, RoBERTa)
  2. WordPiece (BERT)

Byte-pair encoding

- Gage, Philip (1994). "A New Algorithm for Data Compression"

Byte-pair encoding

- Gage, Philip (1994). "A New Algorithm for Data Compression"
- Encode the most common pairs iteratively
  1. look for the most frequent pairing
  2. merge them
  3. repeat (until token or iteration limit)

Byte-pair encoding

- Gage, Philip (1994). "A New Algorithm for Data Compression"
- Encode the most common pairs iteratively
  1. look for the most frequent pairing
  2. merge them
  3. repeat (until token or iteration limit)
- Example (Wikipedia): `aaabdaaabac`

UPPSALA UNIVERSITET

Byte-pair encoding

- Gage, Philip (1994). "A New Algorithm for Data Compression"
- Encode the most common pairs iteratively
  1. look for the most frequent pairing
  2. merge them
  3. repeat (until token or iteration limit)
- Example (Wikipedia): `aaabdaaabac`

  Step 1: `ZabdZabac`
  `Z=aa`

Byte-pair encoding

- Gage, Philip (1994). "A New Algorithm for Data Compression"
- Encode the most common pairs iteratively
  1. look for the most frequent pairing
  2. merge them
  3. repeat (until token or iteration limit)
- Example (Wikipedia): `aaabdaaabac`

  Step 1: `ZabdZabac`
  `Z=aa`

  Step 2: `ZYdZYac`
  `Z=aa`
  `Y=ab`

UPPSALA UNIVERSITET

Byte-pair encoding

- Gage, Philip (1994). "A New Algorithm for Data Compression"
- Encode the most common pairs iteratively
  1. look for the most frequent pairing
  2. merge them
  3. repeat (until token or iteration limit)
- Example (Wikipedia): `aaabdaaabac`

  Step 1: `ZabdZabac`
  `Z=aa`

  Step 2: `ZYdZYac`
  `Z=aa`
  `Y=ab`

  Step 3: `XdXac`
  `Z=aa`
  `Y=ab`
  `X=ZY`

Byte-pair encoding

1. look for the most frequent pairing
2. merge them
3. repeat (until token or iteration limit)

- Example : 9:`text_`, 10:`texting_`,11:`context_`

Byte-pair encoding

1. look for the most frequent pairing
2. merge them
3. repeat (until token or iteration limit)

- Example : 9:`text_`, 10:`texting_`,11:`context_`

  Step 1 (most common: "te"): {`te`}

Byte-pair encoding

1. look for the most frequent pairing
2. merge them
3. repeat (until token or iteration limit)

- Example : 9:`text_`, 10:`texting_`,11:`context_`

  Step 1 (most common: "te"): {`te`}

  ...

  Step i (most common: "text_"): {`text_`}

Byte-pair encoding

1. look for the most frequent pairing
2. merge them
3. repeat (until token or iteration limit)

- Example : 9:`text_`, 10:`texting_`,11:`context_`

  Step 1 (most common: "te"): {`te`}

  ...
  Step i (most common: "text_"): {`text_`}

  ...
  Step j (most common: "con"): {`text_`,con}

Byte-pair encoding

1. look for the most frequent pairing
2. merge them
3. repeat (until token or iteration limit)

- Example : 9:`text_`, 10:`texting_`,11:`context_`

  Step 1 (most common: "te"): {`te`}

  ...
  Step i (most common: "text_"): {`text_`}

  ...
  Step j (most common: "con"): {`text_`,con}

  ...
  Step k (most common: "texting"): {`text_`,con,`texting`}

Byte-pair encoding

1. look for the most frequent pairing
2. merge them
3. repeat (until token or iteration limit)

- Example : 9:`text_`, 10:`texting_`,11:`context_`

  Step 1 (most common: "te"): {`te`}

  ...
  Step i (most common: "text_"): {`text_`}

  ...
  Step j (most common: "con"): {`text_`,con}

  ...
  Step k (most common: "texting"): {`text_`,con,`texting`}

UPPSALA UNIVERSITET

WordPiece

- BPE difficulty: Which pair to choose (if they are approx. equally frequent)?

WordPiece

- BPE difficulty: Which pair to choose (if they are approx. equally frequent)?
- Schuster and Kaisuke (2012) present the WordPiece model

WordPiece

- BPE difficulty: Which pair to choose (if they are approx. equally frequent)?
- Schuster and Kaisuke (2012) present the WordPiece model
- Let $P(i, j)$ be the probability of observing the pair ij and $P(i)$ observing i.

UPPSALA UNIVERSITET

WordPiece

- BPE difficulty: Which pair to choose (if they are approx. equally frequent)?
- Schuster and Kaisuke (2012) present the WordPiece model
- Let $P(i, j)$ be the probability of observing the pair ij and $P(i)$ observing i.
- BPE: Choose highest $P(i, j)$

WordPiece

- BPE difficulty: Which pair to choose (if they are approx. equally frequent)?
- Schuster and Kaisuke (2012) present the WordPiece model
- Let $P(i,j)$ be the probability of observing the pair ij and $P(i)$ observing i.
- BPE: Choose highest $P(i,j)$
- Wordpiece: Choose highest $P(i,j)/(P(i)P(j))$

UPPSALA UNIVERSITET

Section 4

Transformer-Encoder Models

Transformer-Encoder Models

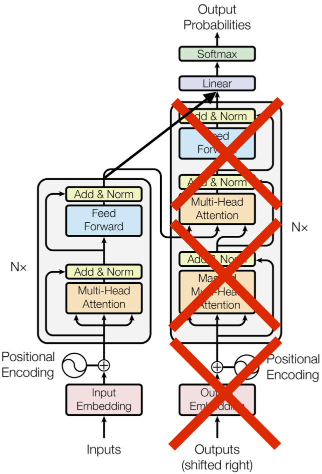

Figure: The Transformer Architecture (Vaswani et al., 2017)

Transformer-Encoder Models

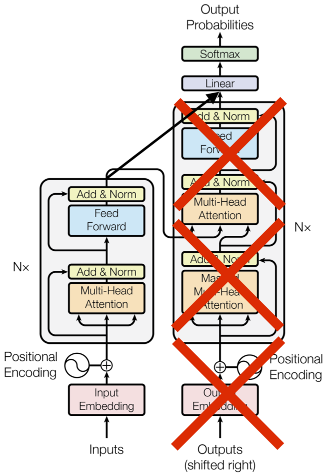

Figure: The Transformer Architecture (Vaswani et al., 2017)

- Common models are BERT and RoBERTa

BERT

- Bidirectional Encoder Representations from Transformers (BERT)
- Introduced in Devlin et al. (2018)

BERT

- Bidirectional Encoder Representations from Transformers (BERT)
- Introduced in Devlin et al. (2018)
- State-of-the-Art in many text prediction tasks, such as
  - Named-Entity Recognition
  - Text Classification

BERT

- Bidirectional Encoder Representations from Transformers (BERT)
- Introduced in Devlin et al. (2018)
- State-of-the-Art in many text prediction tasks, such as
  - Named-Entity Recognition
  - Text Classification
- Many flavors, such as RoBERTa, AlBERT, etc.

BERT

- Bidirectional Encoder Representations from Transformers (BERT)
- Introduced in Devlin et al. (2018)
- State-of-the-Art in many text prediction tasks, such as
  - Named-Entity Recognition
  - Text Classification
- Many flavors, such as RoBERTa, AlBERT, etc.
- Pre-trained on a large corpus

UPPSALA UNIVERSITET

BERT

- Bidirectional Encoder Representations from Transformers (BERT)
- Introduced in Devlin et al. (2018)
- State-of-the-Art in many text prediction tasks, such as
  - Named-Entity Recognition
  - Text Classification
- Many flavors, such as RoBERTa, AlBERT, etc.
- Pre-trained on a large corpus
- Then fine-tuned for a specific problem

BERT

- Bidirectional Encoder Representations from Transformers (BERT)
- Introduced in Devlin et al. (2018)
- State-of-the-Art in many text prediction tasks, such as
  - Named-Entity Recognition
  - Text Classification
- Many flavors, such as RoBERTa, AlBERT, etc.
- Pre-trained on a large corpus
- Then fine-tuned for a specific problem
- Available English, Swedish and many other languages (The National Library)

BERT

- Bidirectional Encoder Representations from Transformers (BERT)
- Introduced in Devlin et al. (2018)
- State-of-the-Art in many text prediction tasks, such as
  - Named-Entity Recognition
  - Text Classification
- Many flavors, such as RoBERTa, AlBERT, etc.
- Pre-trained on a large corpus
- Then fine-tuned for a specific problem
- Available English, Swedish and many other languages (The National Library)
- And again, I rely o Alammar (2018) The illustrated BERT

BERT and transfer learning

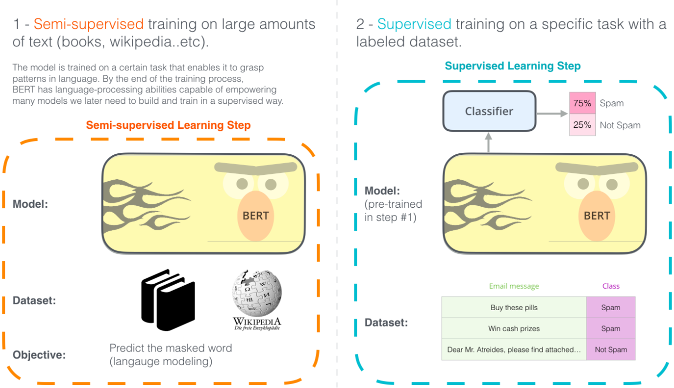

Figure: Using BERT for Transfer Learning (Alammar, 2018b)

The BERT model

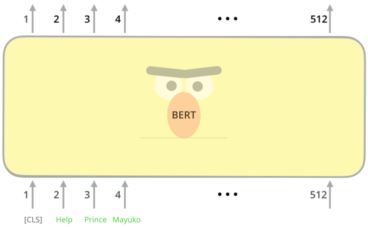

Figure: The BERT model (Alammar, 2018b)

BERT Architecture

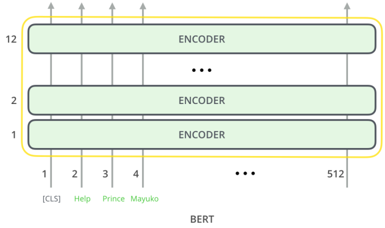

Figure: Opening up BERT (Alammar, 2018b)

Training Task 1: Masked Language Model

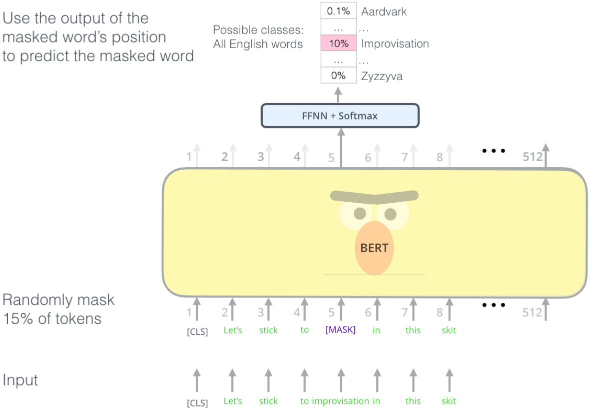

Figure: Masked Language Modeling (Alammar, 2018c)

Training task 2: Next Sentence Prediction

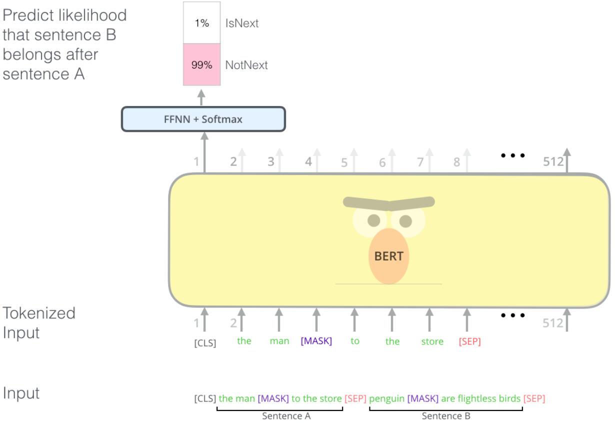

Figure: Next Sentence Prediction (Alammar, 2018c)

Using BERT for Classification

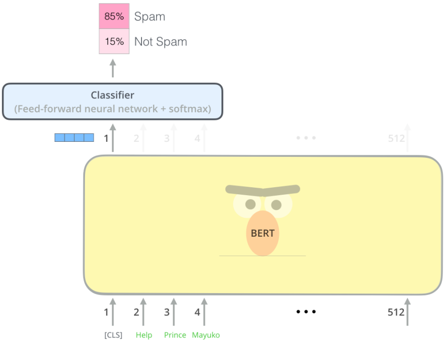

Figure: Using BERT for classification (Alammar, 2018c)

BERT and Contextualized embeddings

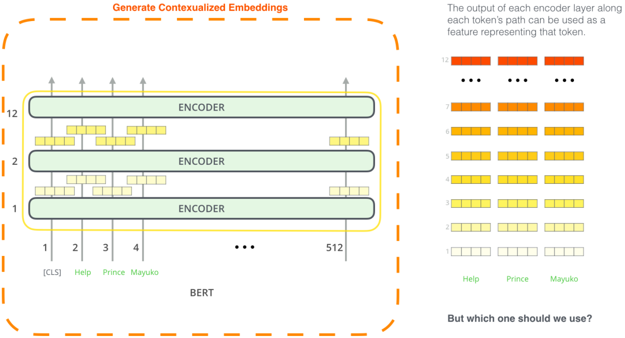

Figure: Contextualized Embeddings (Alammar, 2018c)

Using Contextualized Embeddings

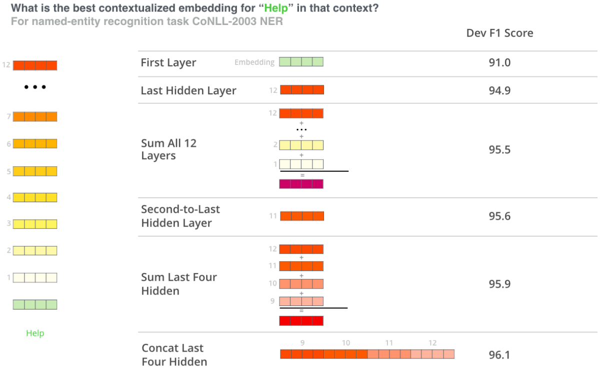

Figure: Using Contextualized Embeddings (Alammar, 2018c)

RoBERTa

- RoBERTa uses a similar model as BERT with some important modifications
  1. Dynamic masking instead of Static masking

RoBERTa

- RoBERTa uses a similar model as BERT with some important modifications
    1. Dynamic masking instead of Static masking
    2. Longer sequences included compared to BERT

RoBERTa

- RoBERTa uses a similar model as BERT with some important modifications
  1. Dynamic masking instead of Static masking
  2. Longer sequences included compared to BERT
  3. No next sentence prediction

RoBERTa

- RoBERTa uses a similar model as BERT with some important modifications
  1. Dynamic masking instead of Static masking
  2. Longer sequences included compared to BERT
  3. No next sentence prediction
  4. Increased vocabulary size

RoBERTa

- RoBERTa uses a similar model as BERT with some important modifications
  1. Dynamic masking instead of Static masking
  2. Longer sequences included compared to BERT
  3. No next sentence prediction
  4. Increased vocabulary size
  5. Trained on more data (160Gb vs. 13 Gb), for longer and with larger batch sizes