Transfer Learning for NLP

COM4513/6513 Natural Language Processing

Nikos Aletras
n.aletras@sheffield.ac.uk

@nikaletras

Computer Science Department

Week 10
Spring 2020

The University Of Sheffield.

In lectures 6 and 8...

- **Neural language modelling:** Probability of a word given some context

In lectures 6 and 8...

- **Neural language modelling:** Probability of a word given some context
    - Feedforward neural networks, e.g. skipgram

In lectures 6 and 8...

- **Neural language modelling:** Probability of a word given some context
  - Feedforward neural networks, e.g. skipgram
  - Recurrent neural networks, e.g. LSTM/GRU

In lectures 6 and 8...

- **Neural language modelling:** Probability of a word given some context
  - Feedforward neural networks, e.g. skipgram
  - Recurrent neural networks, e.g. LSTM/GRU
- Neural LMs are trained on vast amounts of data

In lectures 6 and 8...

- **Neural language modelling:** Probability of a word given some context
  - Feedforward neural networks, e.g. skipgram
  - Recurrent neural networks, e.g. LSTM/GRU
- Neural LMs are trained on vast amounts of data
- Labelled data is cheap, i.e. large publicly available corpora (aka **self supervision**)

In lectures 6 and 8...

- **Neural language modelling:** Probability of a word given some context
  - Feedforward neural networks, e.g. skipgram
  - Recurrent neural networks, e.g. LSTM/GRU
- Neural LMs are trained on vast amounts of data
- Labelled data is cheap, i.e. large publicly available corpora (aka **self supervision**)
- Can we make use of this knowledge in downstream tasks where data might be scarce?

In this lecture...

- **Transfer learning:** Re-use and adapt already pre-trained supervised machine learning models on a target task

In this lecture...

- **Transfer learning:** Re-use and adapt already pre-trained supervised machine learning models on a target task
- How we can re-use and neural LMs on target tasks (e.g. text classification, machine translation, question answering, etc.)

Definition of Transfer Learning

A machine learning approach where models trained on a **source** task (or domain) are adapted to a related **target** task[1] (or domain)

[1] Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. IEEE Transactions on knowledge and data engineering, 22(10), 1345-1359

Definition of Transfer Learning (more formally)

$$\begin{aligned} \text{Domain: } & \mathcal{D} = \{\mathcal{X}, P(X)\} \\ \text{Task: } & \mathcal{T} \text{ where } y \in \mathcal{Y} \\ \text{Cond. Prob. Distrib.: } & P(Y|X) \end{aligned}$$

Given a source domain $\mathcal{D}_S$ and a corresponding task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and task $\mathcal{T}_T$, learn a new model that computes the target conditional probability distribution $P(Y_T|X_T)$ in $\mathcal{D}_T$ given information from $\mathcal{D}_S$ and $\mathcal{T}_S$

Transfer Learning Variants

- $\mathcal{X}_S \neq \mathcal{X}_T$: Different feature spaces in source and target domains, e.g. documents written in different languages (**cross-lingual adaptation**)

Transfer Learning Variants

- $\mathcal{X}_S \neq \mathcal{X}_T$: Different feature spaces in source and target domains, e.g. documents written in different languages (**cross-lingual adaptation**)
- $P(X_S) \neq P(X_T)$: Different marginal probability distributions in source and target domains, e.g. restaurant reviews vs electronic product reviews (**domain adaptation**)

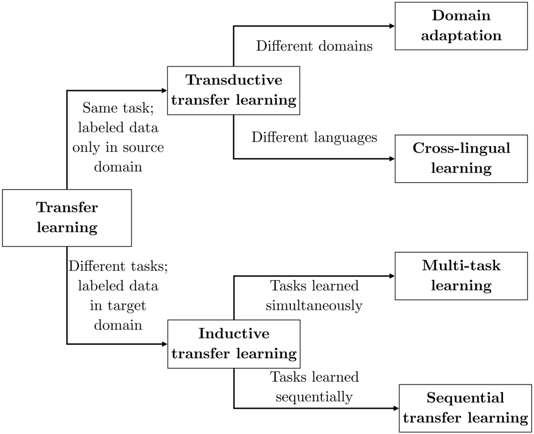
Transfer Learning Variants

- $\mathcal{X}_S \neq \mathcal{X}_T$: Different feature spaces in source and target domains, e.g. documents written in different languages (**cross-lingual adaptation**)
- $P(X_S) \neq P(X_T)$: Different marginal probability distributions in source and target domains, e.g. restaurant reviews vs electronic product reviews (**domain adaptation**)
- $\mathcal{Y}_S \neq \mathcal{Y}_T$: Different tasks (label sets), e.g. LM as source task and sentiment analysis as target task

Transfer Learning Variants

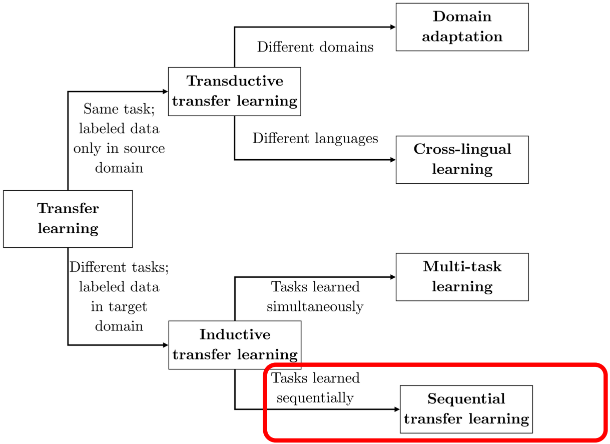
- $\mathcal{X}_S \neq \mathcal{X}_T$: Different feature spaces in source and target domains, e.g. documents written in different languages (**cross-lingual adaptation**)
- $P(X_S) \neq P(X_T)$: Different marginal probability distributions in source and target domains, e.g. restaurant reviews vs electronic product reviews (**domain adaptation**)
- $\mathcal{Y}_S \neq \mathcal{Y}_T$: Different tasks (label sets), e.g. LM as source task and sentiment analysis as target task
- $P(Y_S|X_S) \neq P(Y_T|X_T)$: Different conditional probability distributions between source and target tasks, e.g. source and target documents are unbalanced regarding to their classes

Transfer Learning Taxonomy[2]



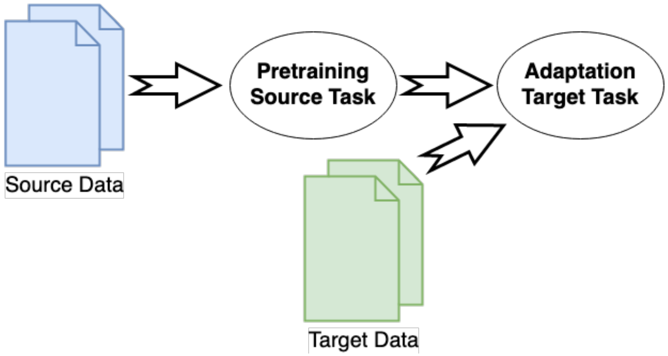
[2] Ruder, S. (2019). Neural transfer learning for natural language processing (Doctoral dissertation, NUI Galway)

Transfer Learning Taxonomy[2]



[2] Ruder, S. (2019). Neural transfer learning for natural language processing (Doctoral dissertation, NUI Galway)

Sequential Transfer Learning



Pretraining: Language Modelling

- Source task: Language modelling

Pretraining: Language Modelling

- Source task: Language modelling
- Cheap: no human annotations required

Pretraining: Language Modelling

- Source task: Language modelling
- Cheap: no human annotations required
- Large amounts of publicly available data: Wikipedia, Web etc.

Pretraining: Language Modelling

- Source task: Language modelling
- Cheap: no human annotations required
- Large amounts of publicly available data: Wikipedia, Web etc.
- Learn both word and sentence representations

Pretraining: Language Modelling

- Source task: Language modelling
- Cheap: no human annotations required
- Large amounts of publicly available data: Wikipedia, Web etc.
- Learn both word and sentence representations
- Many variants in objective functions:

Pretraining: Language Modelling

- Source task: Language modelling
- Cheap: no human annotations required
- Large amounts of publicly available data: Wikipedia, Web etc.
- Learn both word and sentence representations
- Many variants in objective functions:
    - predict all context words given target word (and vice versa)

Pretraining: Language Modelling

- Source task: Language modelling
- Cheap: no human annotations required
- Large amounts of publicly available data: Wikipedia, Web etc.
- Learn both word and sentence representations
- Many variants in objective functions:
  - predict all context words given target word (and vice versa)
  - predict masked context words (fill-in-the-blank)

Pretraining: Language Modelling

- Source task: Language modelling
- Cheap: no human annotations required
- Large amounts of publicly available data: Wikipedia, Web etc.
- Learn both word and sentence representations
- Many variants in objective functions:
  - predict all context words given target word (and vice versa)
  - predict masked context words (fill-in-the-blank)
  - predict perturbed context

Pretraining: Language Modelling

- Source task: Language modelling
- Cheap: no human annotations required
- Large amounts of publicly available data: Wikipedia, Web etc.
- Learn both word and sentence representations
- Many variants in objective functions:
  - predict all context words given target word (and vice versa)
  - predict masked context words (fill-in-the-blank)
  - predict perturbed context
- Models?

Pretraining: Models

- Feedforward networks, e.g. word2vec[3]

[3] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119)

[4] Howard, J., & Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (pp. 328-339).

[5] Vaswani, Ashish, et al. (2017) "Attention is all you need." Advances in neural information processing systems.

[6] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies(pp. 4171-4186).

Pretraining: Models

- Feedforward networks, e.g. word2vec[3]
- LSTM, e.g. Universal Language Model Fine-tuning (ULMFiT[4])

[3] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119)

[4] Howard, J., & Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (pp. 328-339).

[5] Vaswani, Ashish, et al. (2017) "Attention is all you need." Advances in neural information processing systems.

[6] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies(pp. 4171-4186).

Pretraining: Models

- Feedforward networks, e.g. word2vec[3]
- LSTM, e.g. Universal Language Model Fine-tuning (ULMFiT[4])
- Transformer[5] Network, e.g. Bidirectional Encoder Representations from Transformers (BERT[6])

[3] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119)

[4] Howard, J., & Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (pp. 328-339).

[5] Vaswani, Ashish, et al. (2017) "Attention is all you need." Advances in neural information processing systems.

[6] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies(pp. 4171-4186).

BERT: Pre-training of Deep Bidirectional Transformers

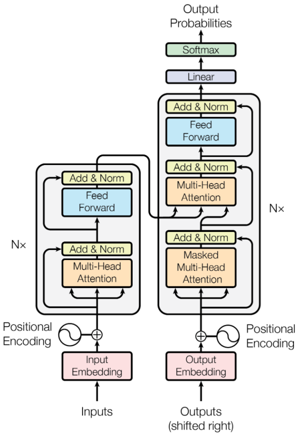


Figure 1: The Transformer - model architecture.

- Encoder 6 layers: 2 sub-layers each

BERT: Pre-training of Deep Bidirectional Transformers

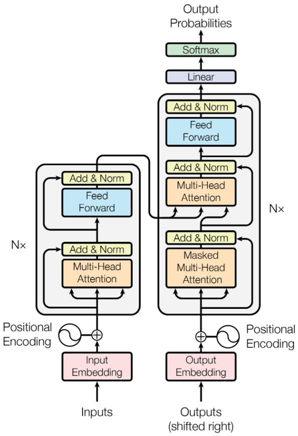


Figure 1: The Transformer - model architecture.

- Encoder 6 layers: 2 sub-layers each
- Sub-layer 1: Multi-head self-attention mechanism (Lectures 7-8)

BERT: Pre-training of Deep Bidirectional Transformers

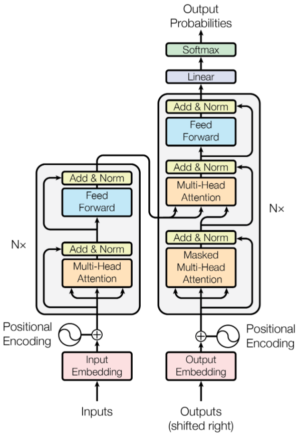


Figure 1: The Transformer - model architecture.

- Encoder 6 layers: 2 sub-layers each
- Sub-layer 1: Multi-head self-attention mechanism (Lectures 7-8)
- Sub-layer 2: Position-wise fully connected layer

BERT: Pre-training of Deep Bidirectional Transformers

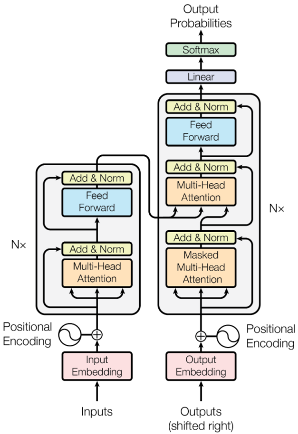


Figure 1: The Transformer - model architecture.

- Encoder 6 layers: 2 sub-layers each
- Sub-layer 1: Multi-head self-attention mechanism (Lectures 7-8)
- Sub-layer 2: Position-wise fully connected layer
- Output of each sublayer is combined with its input followed by layer norm

BERT: Pre-training of Deep Bidirectional Transformers

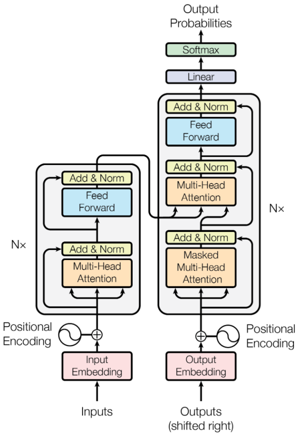


Figure 1: The Transformer - model architecture.

- Encoder 6 layers: 2 sub-layers each
- Sub-layer 1: Multi-head self-attention mechanism (Lectures 7-8)
- Sub-layer 2: Position-wise fully connected layer
- Output of each sublayer is combined with its input followed by layer norm
- Input is combined with a positional embedding (containing information for particular position in the sequence)

BERT: Pre-training of Deep Bidirectional Transformers

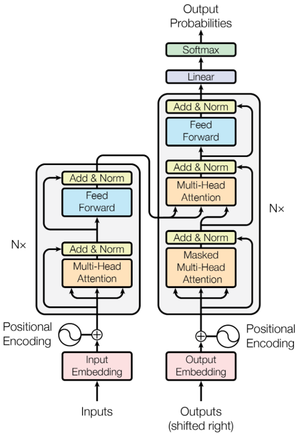


Figure 1: The Transformer - model architecture.

- Decoder 6 layers: 3 sub-layers each

BERT: Pre-training of Deep Bidirectional Transformers

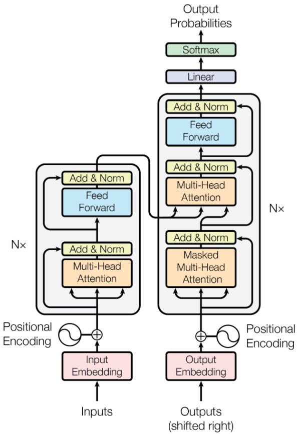


Figure 1: The Transformer - model architecture.

- Decoder 6 layers: 3 sub-layers each
- Sub-layer 3: multihead attention over the output of the encoder

BERT: Pre-training of Deep Bidirectional Transformers

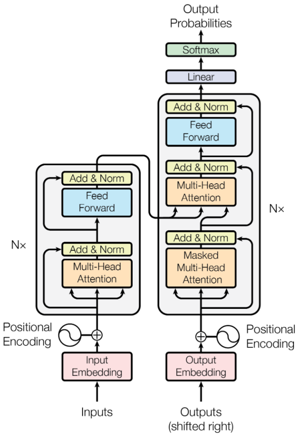


Figure 1: The Transformer - model architecture.

- Decoder 6 layers: 3 sub-layers each
- Sub-layer 3: multihead attention over the output of the encoder
- Output layer

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:
  - Masked LM

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:
  - Masked LM
  - Next sentence prediction

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:
  - Masked LM
  - Next sentence prediction
- Input:

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:
  - Masked LM
  - Next sentence prediction
- Input:
  - WordPiece embeddings

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:
  - Masked LM
  - Next sentence prediction
- Input:
  - WordPiece embeddings
  - $[\mathrm{CLS}]$ (special classification token) is appended at the beginning of each sequence

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:
  - Masked LM
  - Next sentence prediction
- Input:
  - WordPiece embeddings
  - $[\mathrm{CLS}]$ (special classification token) is appended at the beginning of each sequence
  - Sentences are separated wirh a special token $[\mathrm{SEP}]$

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:
  - Masked LM
  - Next sentence prediction
- Input:
  - WordPiece embeddings
  - $[\mathrm{CLS}]$ (special classification token) is appended at the beginning of each sequence
  - Sentences are separated wirh a special token $[\mathrm{SEP}]$
  - Token input representation: summing token, segmentation and position embeddings

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:
  - Masked LM
  - Next sentence prediction
- Input:
  - WordPiece embeddings
  - $[\mathrm{CLS}]$ (special classification token) is appended at the beginning of each sequence
  - Sentences are separated wirh a special token $[\mathrm{SEP}]$
  - Token input representation: summing token, segmentation and position embeddings
- The final hidden state corresponding to $[\mathrm{CLS}]$ token is used as the aggregate sequence representation for classification tasks (e.g. target tasks)

BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:
  - Masked LM
  - Next sentence prediction
- Input:
  - WordPiece embeddings
  - $[\mathrm{CLS}]$ (special classification token) is appended at the beginning of each sequence
  - Sentences are separated wirh a special token $[\mathrm{SEP}]$
  - Token input representation: summing token, segmentation and position embeddings
- The final hidden state corresponding to $[\mathrm{CLS}]$ token is used as the aggregate sequence representation for classification tasks (e.g. target tasks)
- BERT variants: XLNet, RoBERTa, ALBERT

Adaptation

- Initialise your encoder on the target task using the weights you learned in LM

Adaptation

- Initialise your encoder on the target task using the weights you learned in LM
- Change the output layer of your network to match the target task

Adaptation

- Initialise your encoder on the target task using the weights you learned in LM
- Change the output layer of your network to match the target task
- Freeze the weights of the pretrained word embeddings/encoder

Adaptation

- Initialise your encoder on the target task using the weights you learned in LM
- Change the output layer of your network to match the target task
- Freeze the weights of the pretrained word embeddings/encoder
- Learn the weights of the output layer on the target task data

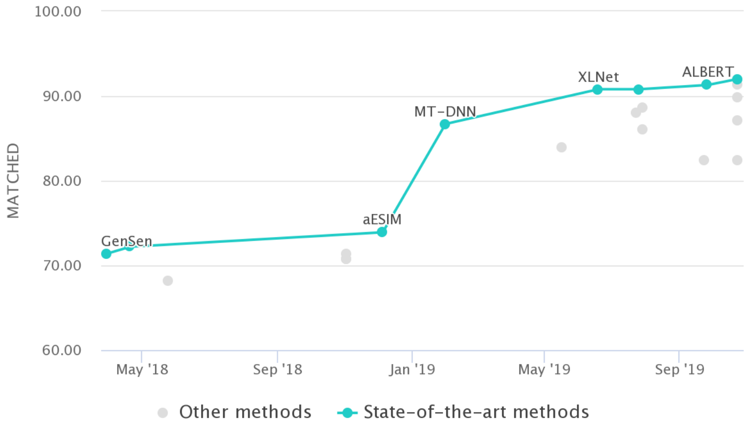
Adaptation

- Initialise your encoder on the target task using the weights you learned in LM
- Change the output layer of your network to match the target task
- Freeze the weights of the pretrained word embeddings/encoder
- Learn the weights of the output layer on the target task data
- Unfreeze the weights of the pretrained components and fine-tune them (additional training steps with very small learning rate)

Adaptation

- Initialise your encoder on the target task using the weights you learned in LM
- Change the output layer of your network to match the target task
- Freeze the weights of the pretrained word embeddings/encoder
- Learn the weights of the output layer on the target task data
- Unfreeze the weights of the pretrained components and fine-tune them (additional training steps with very small learning rate)
- In ULMFiT, the LM encoder (LSTM) is fine-tuned on the test data before adaptation

Does it work?



Performance on Natural Language Inference on MultiNLI[7]

[7] https://paperswithcode.com/sota/natural-language-inference-on-multinli

Bibliography

- Blog post on Transfer Learning by S. Ruder
- Blog post on Transfer Learning in NLP by S. Ruder
- Blog post on BERT by Samia

Thanks!