Transfer Learning for NLP
COM4513/6513 Natural Language Processing

Nikos Aletras
n.aletras@sheffield.ac.uk
@nikaletras

Computer Science Department

Week 10
Spring 2020
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

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?
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.)
A machine learning approach where models trained on a source task (or domain) are adapted to a related target task\(^1\) (or domain)

Definition of Transfer Learning (more formally)

Domain: \( D = \{ \mathcal{X}, P(X) \} \)

Task: \( T \) where \( y \in \mathcal{Y} \)

Cond. Prob. Distrib.: \( P(Y|X) \)

Given a source domain \( D_S \) and a corresponding task \( T_S \), a target domain \( D_T \) and task \( T_T \), learn a new model that computes the target conditional probability distribution \( P(Y_T|X_T) \) in \( D_T \) given information from \( D_S \) and \( T_S \)
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)
- $P(\mathcal{X}_S) \neq P(\mathcal{X}_T)$: 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)
- $P(\mathcal{X}_S) \neq P(\mathcal{X}_T)$: 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
- $P(\mathcal{Y}_S|\mathcal{X}_S) \neq P(\mathcal{Y}_T|\mathcal{X}_T)$: Different conditional probability distributions between source and target tasks, e.g. source and target documents are unbalanced regarding to their classes
Transfer Learning Taxonomy

Transductive transfer learning
- Same task; labeled data only in source domain

Inductive transfer learning
- Different tasks; labeled data in target domain

Domain adaptation
- Different domains

Cross-lingual learning
- Different languages

Multi-task learning
- Tasks learned simultaneously

Sequential transfer learning
- Tasks learned sequentially

---

Transfer Learning Taxonomy

- **Transfer learning**
  - Same task; labeled data only in source domain
  - Different tasks; labeled data in target domain
  - Different domains
  - Different languages

- **Transductive transfer learning**
  - Tasks learned sequentially

- **Inductive transfer learning**
  - Tasks learned simultaneously

- **Domain adaptation**

- **Cross-lingual learning**

- **Multi-task learning**

- **Sequential transfer learning**

---

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?
- Feedforward networks, e.g. word2vec

---


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

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).


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

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)


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

BERT: Pre-training of Deep Bidirectional Transformers

- Encoder 6 layers: 2 sub-layers each

![Diagram of the Transformer model architecture.](image)

Figure 1: The Transformer - model architecture.
BERT: Pre-training of Deep Bidirectional Transformers

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

Figure 1: The Transformer - model architecture.
BERT: Pre-training of Deep Bidirectional Transformers

- 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

Figure 1: The Transformer - model architecture.
BERT: Pre-training of Deep Bidirectional Transformers

- 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

Figure 1: The Transformer - model architecture.
BERT: Pre-training of Deep Bidirectional Transformers

- 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)

Figure 1: The Transformer - model architecture.
BERT: Pre-training of Deep Bidirectional Transformers

- Decoder 6 layers: 3 sub-layers each

Figure 1: The Transformer - model architecture.
BERT: Pre-training of Deep Bidirectional Transformers

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

Figure 1: The Transformer - model architecture.
BERT: Pre-training of Deep Bidirectional Transformers

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

Figure 1: The Transformer - model architecture.
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 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
  - [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
  - [CLS] (special classification token) is appended at the beginning of each sequence
  - Sentences are separated with a special token [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
  - [CLS] (special classification token) is appended at the beginning of each sequence
  - Sentences are separated with a special token [SEP]
  - Token input representation: summing token, segmentation and position embeddings

The final hidden state corresponding to [CLS] token is used as the aggregate sequence representation for classification tasks (e.g. target tasks).

BERT variants: XLNet, RoBERTa, ALBERT
BERT: Pre-training of Deep Bidirectional Transformers

- BERT is a bidirectional Transformer
- Pretrained on two tasks:
  - Masked LM
  - Next sentence prediction
- Input:
  - WordPiece embeddings
  - [CLS] (special classification token) is appended at the beginning of each sequence
  - Sentences are separated with a special token [SEP]
  - Token input representation: summing token, segmentation and position embeddings
- The final hidden state corresponding to [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
  - [CLS] (special classification token) is appended at the beginning of each sequence
  - Sentences are separated with a special token [SEP]
  - Token input representation: summing token, segmentation and position embeddings
- The final hidden state corresponding to [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

In ULMFiT, the LM encoder (LSTM) is fine-tuned on the test data before adaptation.
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

In ULMFiT, the LM encoder (LSTM) is fine-tuned on the test data before adaptation
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

In ULMFiT, the LM encoder (LSTM) is fine-tuned on the test data before adaptation
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

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!