Transformer and Unsupervised NMT
Erweiterungsmodul: Machine Translation
Sommersemester 2020

Dario Stojanovski

Center for Information and Language Processing
Ludwig Maximilian University of Munich

dario@cis.lmu.de

June 9, 2020
Outline

1. Transformer
   - Self-attention
   - Transformer building blocks

2. Document-Level Neural Machine Translation
   - Discourse-level phenomena
   - Models

3. Unsupervised Neural Machine Translation
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
1. **Transformer**
   - Self-attention
   - Transformer building blocks

2. **Document-Level Neural Machine Translation**
   - Discourse-level phenomena
   - Models

3. **Unsupervised Neural Machine Translation**
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
A Brief History

- Machine translation: the task of receiving a sentence in the source language and outputting a translation in the target language

- RNN encoder-decoder proved that this is possible with neural networks (Sutskever, Vinyals, and Le 2014)

- Attention-based encoder-decoder provided state-of-the-art performance (Bahdanau, Cho, and Bengio 2015)

- Convolutional encoder-decoder (Gehring et al. 2017)

- Transformer - Attention is all you need (Vaswani et al. 2017)
Why Not RNN Encoder-Decoder

- Up to Transformer, most models implemented with attention-based RNN encoder-decoder
- RNNs process words sequentially
- They do not lend themselves to parallelization
- Difficult to model long-term dependencies
- Bidirectional RNN enable more meaningful modeling of words near the end

- Attention enables direct access to all hidden states

- Still not a fundamental solution to the problem, the interplay between hidden states of arbitrary words is limited
Parallelization

- Very important in the big data era
- Robust MT requires millions of parallel sentences
- Convolutional encoder-decoder help with parallelization, but require many layers to be able to model long-term dependencies
First purely self-attention-based encoder-decoder model - all hidden states interact with each other

Can easily be parallelized

State-of-the-art model architecture for MT and most NLP tasks

Used for training large Language Models (BERT, GPT-2) which provide large improvements in many NLP tasks by making use of transfer learning
1. **Transformer**  
   - Self-attention  
   - Transformer building blocks

2. **Document-Level Neural Machine Translation**  
   - Discourse-level phenomena  
   - Models

3. **Unsupervised Neural Machine Translation**  
   - Initialization  
   - Denoising auto-encoding  
   - Iterative backtranslation  
   - Combining the techniques
- Hidden states computed with an RNN
- Attention computed at each decoder step
- Attention over encoder hidden states
- Computed “context” vector integrated into decoder hidden state

Figure from Bahdanau, Cho, and Bengio 2015
Attention in RNN Encoder-Decoder

- Motivation: do not summarize the whole sentence in a single vector
- Variable length representation - keep RNN hidden states for each word
- Pay more attention to more relevant hidden states (words)
- Attention determines what source words are important for predicting the next target word

Figure from Bahdanau, Cho, and Bengio 2015
Self-Attention

- Also intra-attention - relating different positions of the same input sequence

- All hidden states are computed using only attention
  - No recurrence - all recurrent connections replaced with attention
  - Self-attention replaces the RNN

- A given hidden state is a weighted linear combination of all hidden states
  - A given word interacts with all other words, including itself
Self-Attention

The cat chased the mouse

Dario Stojanovski (LMU Munich)
Self-Attention

The cat chased the mouse.
Self-Attention

The cat chased the mouse

$w_{13}$ $w_{23}$ $w_{33}$ $w_{43}$ $w_{53}$
Self-Attention

The cat chased the mouse

w_{14} \quad w_{24} \quad w_{34} \quad w_{44} \quad w_{54}
Self-Attention

The cat chased the mouse

Dario Stojanovski (LMU Munich)
1. **Transformer**
   - Self-attention
   - Transformer building blocks

2. **Document-Level Neural Machine Translation**
   - Discourse-level phenomena
   - Models

3. **Unsupervised Neural Machine Translation**
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
Queries $Q$, Keys $K$, Values $V$

The interaction between $Q$ and $K$ determines how to score $V$ (how important certain values are).

In Bahdanau, Cho, and Bengio 2015, $Q$ is the current decoder state, $K$ and $V$ are all encoder states.
Bahdanau, Cho, and Bengio 2015 used multi-layer perceptron for the attention

\[ c_j = \sum_{i}^{T_x} \alpha_{ij} h_i \]

\[ \alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{kj})} \]

\[ e_{ij} = v^T_a \tanh(U_a s'_j + W_a h_i) \]

But it requires additional parameters \( U_a, W_a \) and \( v_a \)

\( s_j \) - decoder hidden state, \( h_i \) - encoder hidden state
Dot-product attention does not require parameters

\[ a(q, k) = qk^T \]

Dimensionality of \( q \) and \( k \) has to be the same.

The scale of the dot product depends on the dimensionality of the vectors. Scale grows as dimensionality grows. Can be fixed by scaling proportionally to the dimensionality:

\[ a(q, k) = \frac{qk^T}{\sqrt{|k|}} \]
Scaled Dot-Product Attention in Transformer

\[
\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V
\]

Vaswani et al. 2017
for self-attention, the input is Q, K and V at the same time

At Transformer encoder layer 1 (E is token embedding)

In practice, the Transformer learns 3 separate projection layers at each encoder or decoder layer for Q, K and V.

\[ Q = EW_1^Q, K = EW_1^K, V = EW_1^V \]
Multi-Head Attention

- Computing one attention may be too brittle

- Separation of concerns - learn multiple separate attentions.

- Intuition is that different attention heads learn different things and are allowed to focus on different things in the input

- Conceptually, one head may learn to incorporate local information for each token representation and another may learn to look for potential long-term dependencies

- Many works try to interpret attention, but it is not clear if strong conclusions can be made by looking at attention scores
Transformer learns $h$ attention heads (initially 8, but lots of other values have been tried).

Linear projection layers $W$ project the input to dimensionality $d/h$. Separate linear layers for each head.

Output of all attention heads is later concatenated and an additional linear layer is applied.

Computational overhead of multiple attention heads compensated by applying attention over smaller parameters spaces.

Vaswani et al. 2017
Multi-Head Attention - Visualizations

http://jalammar.github.io/illustrated-transformer/
Multi-Head Attention - Visualizations

http://jalammar.github.io/illustrated-transformer/
Positional Information

- Positional information is built-in in RNN encoder-decoder by way of processing the input sequentially.

- Self-attention is ignorant of any positional information and therefore has to be explicitly provided.

- Learns separate embeddings for positional information - maximum length has to be specified.

- Transformer introduced sinusoidal positional embeddings - may allow for easier learning of relative position information and extrapolation to unseen sequence lengths.
Transformer

- Multiple encoder and decoder layers
- Token-level and positional embeddings
- Multi-head attention
  - Self-Attention
  - Encoder attention
- Feed-forward networks
- Residual connections (Add)
- Layer normalization (Norm)
- Final linear projection to size of vocabulary and softmax to determine most probable next output token
Transformer Encoder

- Apply multi-head self-attention to input as we talked about before
- Residual connection, add the input to multi-head attention to its output
- Apply layer normalization
- Apply a feed-forward neural network
- Residual connection + layer normalization
Transformer Decoder

- Masked multi-head self-attention
- Attention over encoder hidden states
- Remaining components used as before
Deep neural networks can be difficult to train - gradients may explode or vanish.

Residual connections and layer normalization address these issues.

Feed-forward neural network is applied to each position separately and identically.
Masking

http://jalammar.github.io/illustrated-gpt2/
Comparison to RNN and Convolution

<table>
<thead>
<tr>
<th>Layer Type</th>
<th>Complexity per Layer</th>
<th>Sequential Operations</th>
<th>Maximum Path Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Attention</td>
<td>$O(n^2 \cdot d)$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Recurrent</td>
<td>$O(n \cdot d^2)$</td>
<td>$O(n)$</td>
<td>$O(n)$</td>
</tr>
<tr>
<td>Convolutional</td>
<td>$O(k \cdot n \cdot d^2)$</td>
<td>$O(1)$</td>
<td>$O(\log_k(n))$</td>
</tr>
<tr>
<td>Self-Attention (restricted)</td>
<td>$O(r \cdot n \cdot d)$</td>
<td>$O(1)$</td>
<td>$O(n/r)$</td>
</tr>
</tbody>
</table>

- $n$ - sequence length, $d$ - representation dimension, $k$ - kernel size of convolutions

- Sequential operations - how much can be parallelized

- Maximum Path Length: path length between long-range dependencies
Conclusion

- Self-attention - a powerful mechanism
- Multi-head attention
- Parallelizable, instant access to all inputs, less complexity
- Transformers are ubiquitous in NMT and NLP
1 Transformer
   - Self-attention
   - Transformer building blocks

2 Document-Level Neural Machine Translation
   - Discourse-level phenomena
   - Models

3 Unsupervised Neural Machine Translation
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
Traditionally MT works on the sentence-level - single sentence input and output

Phrase-based and Neural MT are both largely ignorant of broader discourse-level phenomena
Why do we do sentence-level NMT?

- Resource limitations - memory requirements
- Document-level models may be challenging to train
- Document-aligned data is scarce
- A high portion of publicly available datasets either do not preserve the sequential order or lack document boundaries

Why is this the case? - most of textual data is part of some document in one form or another

- Likely because of historical reasons - small number of works on document-level MT
Some works have claimed that they have achieved human parity in MT Hassan et al. 2018

These claims have been challenged by Läubli, Sennrich, and Volk 2018; Toral et al. 2018

Among other remarks, both works note that the manual evaluation was done in a context-agnostic way
1. **Transformer**
   - Self-attention
   - Transformer building blocks

2. **Document-Level Neural Machine Translation**
   - Discourse-level phenomena
   - Models

3. **Unsupervised Neural Machine Translation**
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
Discourse-level phenomena

- Coreference (anaphora) resolution
  - Important for gender and number pronoun agreement
  - *It presents a problem.* → *Er [masculine] präsentiert ein Problem.*
  - Context: *Let me summarize the novel for you.*

- Coherence - consistency to concepts and world knowledge

- Cohesion - consistency to surface formulations

- Deixis - words and phrases that cannot be understood without context (time and place dependent), formality

- Style - “is anybody hurt” and “is someone wounded”

- Domain - important for domain-dependent ambiguous translations
1. Transformer
   - Self-attention
   - Transformer building blocks

2. Document-Level Neural Machine Translation
   - Discourse-level phenomena
   - Models

3. Unsupervised Neural Machine Translation
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
Document-level NMT

- Also referred to as context-aware NMT
- Using contextual (document) information

Main questions:
- How do we process/model the context?
- How do we integrate contextual information in our models?
Document-level MT

- How do context-aware (document-level) models differ?

- **Source-Target**
  - Do they use source or target side contextual information?

- **Fine-grained vs. coarse-grained**
  - Do they aim to obtain a fine-grained or coarse-grained document representation?

- **Input-Output**
  - Do they only input large pieces of text or do they output them as well?

- **Previous-Subsequent context**

- Many methods are a mix of different types.
Concatenation models

- First work by Tiedemann and Scherrer 2017
- Concatenate consecutive sentences on the input side
  - And optionally on the output side as well

**Input**
Let me summarize the novel for you. [SEP] It presents a problem.

**Output**
Ich fasse den Roman für dich zusammen. [SEP] Er präsentiert ein Problem.

- No need to modify the baseline NMT architecture or to come up with a context integration scheme
- Works fine with a limited number of contextual sentences, but some effort is required to scale to a large sized context. Straightforward application to very large sequences almost impossible.
Context integration

- Gating

- $dm_i$ - main sentence representation in decoder (after dec-to-enc attention)

- $dc_i$ - context sentence representation in decoder

- Compute a gate $z_i$

- Control how much contextual information is needed for the prediction of the next word

\[
    z_i = \sigma(U_z dm_i + W_z dc_i)
\]

\[
    d_i = z_i \ast dm_i + (1 - z_i) \ast dc_i
\]
Hierarchical attention

- Miculicich et al. 2018 - attention over tokens followed by attention over sentences
Are context-aware models going to be more widely used by the MT community?

The potential benefits from contextual information are clear

Flexible, simple and scalable models

Transformer LM capable of large input modeling - adaptation to NMT?
1. **Transformer**
   - Self-attention
   - Transformer building blocks

2. **Document-Level Neural Machine Translation**
   - Discourse-level phenomena
   - Models

3. **Unsupervised Neural Machine Translation**
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
NMT models require parallel data to be efficiently trained

Parallel data is scarce across many language pairs and domains

Monolingual data on the other hand is prevalent

There are works making use of monolingual data, but still need parallel data as well

Unsupervised NMT uses monolingual data only
Critical Components of Unsupervised MT

- **Meaningful initialization:**
  - random initialization works for models trained on parallel data, but it is useless for UMT
  - Unsupervised Bilingual Word Embeddings

- **Iterative improvement**
  - Denoising auto-encoding
  - Iterative backtranslation
Unsupervised NMT - Important Ingredients

- Proper initialization - unsupervised BWEs

- Learn to generate proper language
  - Denoising auto-encoding

- Enable continuous improvement
  - Iterative backtranslation

- One model for both translation directions
1. Transformer
   - Self-attention
   - Transformer building blocks

2. Document-Level Neural Machine Translation
   - Discourse-level phenomena
   - Models

3. Unsupervised Neural Machine Translation
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
Unsupervised Bilingual Word Embeddings

- Works well for closely related languages, less so for distant languages
- Use unsupervised BWEs to initialize the NMT embeddings
- Use the unsupervised BWEs to do bilingual lexicon induction
- Use induced lexicon to create word-by-word translation
  - Not appropriate for: reordering, compound words, phrases

Lample et al. 2018c
1 Transformer
   - Self-attention
   - Transformer building blocks

2 Document-Level Neural Machine Translation
   - Discourse-level phenomena
   - Models

3 Unsupervised Neural Machine Translation
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
Denoising Auto-Encoding

- How can we make the model generate proper target language sentences?
- Reconstruct the input - a trivial task because it only involves copying
- What if we add noise to the input and try to reconstruct the original?

Denoising Auto-Encoding - Noise

- Randomly drop words

\[ X = \text{I am tired because I went on a run} \]
\[ \text{Drop}(X) = \text{I am because I went on run} \]

Denoising auto-encoding
\[ \text{Drop}(X) \rightarrow X \]
\[ \text{I am because I went on run} \rightarrow \text{I am tired because I went on a run} \]

- a word is dropped with a certain probability \( p_{wd} \)

- the model learns important properties about the language

- word insertions: \textit{a run}

- perhaps something about semantics as well: \textit{tired - run}
Denosing Auto-Encoding - Noise

- Permute words within a certain neighborhood

\[ X = \text{I want to play tennis on Friday} \]
\[ \text{Shuffle}(X) = \text{I want to on Friday tennis play} \]

Denoising auto-encoding

\[ \text{Shuffle}(X) \rightarrow X \]
\[ \text{I want to on Friday tennis play} \rightarrow \text{I want to play tennis on Friday} \]

In German

Ich möchte am Freitag Tennis spielen

- the model can learn to do word reordering
- important when word order between the 2 languages is not the same, e.g. English and German
1 Transformer
   - Self-attention
   - Transformer building blocks

2 Document-Level Neural Machine Translation
   - Discourse-level phenomena
   - Models

3 Unsupervised Neural Machine Translation
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
Backtranslation

- Effective way to integrate monolingual data

Steps:
- Find target language monolingual data $M_t$
- Train a reverse NMT model - if we care about English to German translation, we will train an additional German to English model $NMT_{de-en}$
- Use the trained reverse model $NMT_{de-en}$ to translate the monolingual data $M_t$
- Create a pseudo parallel corpus: pair the corresponding monolingual sentences with the obtained translations
- Translations on the source side, monolingual data on the target
- Fine-tune the $NMT_{en-de}$ model with the new pseudo parallel corpus
I think, therefore I am

Entities should not be multiplied unnecessarily

The limits of my language mean the limits of my world
Iterative Backtranslation

- Backtranslation improves performances
  - Translations are noisy - encoder learns to deal with noise better
  - Almost arbitrary amount of pseudo parallel data

- But why should we do it only once?

- If the model improves, we can use the improved model to generate new backtranslations and repeat the process - iterative backtranslation

- On-the-fly iterative backtranslation
  - For UNMT, one model can do both translation directions
  - No need to backtranslate the whole corpus before fine-tuning
  - Backtranslate and fine-tune alternately
1. Transformer
   - Self-attention
   - Transformer building blocks

2. Document-Level Neural Machine Translation
   - Discourse-level phenomena
   - Models

3. Unsupervised Neural Machine Translation
   - Initialization
   - Denoising auto-encoding
   - Iterative backtranslation
   - Combining the techniques
Putting It All Together

\[ x_{src} \sim D_{src} \rightarrow C_{\text{noise}} \rightarrow e(\cdot, \text{src}) \text{ encoder} \rightarrow z_{src} \rightarrow d(\cdot, \text{src}) \text{ decoder} \rightarrow \mathcal{L}_{\text{auto}} \]

\[ x_{tgt} \sim D_{tgt} \rightarrow C_{\text{noise}} \rightarrow e(\cdot, \text{tgt}) \text{ encoder} \rightarrow z_{tgt} \rightarrow d(\cdot, \text{tgt}) \text{ decoder} \rightarrow \mathcal{L}_{\text{auto}} \]

\[ x_{tgt} \sim D_{tgt} \rightarrow M_{\text{model at previous iter}} \rightarrow y_{src} \rightarrow C_{\text{noise}} \rightarrow e(\cdot, \text{src}) \text{ encoder} \rightarrow z_{src} \rightarrow d(\cdot, \text{src}) \text{ decoder} \rightarrow \mathcal{L}_{\text{cd}} \]

\[ x_{src} \sim D_{src} \rightarrow M_{\text{model at previous iter}} \rightarrow y_{tgt} \rightarrow C_{\text{noise}} \rightarrow e(\cdot, \text{tgt}) \text{ encoder} \rightarrow z_{tgt} \rightarrow d(\cdot, \text{tgt}) \text{ decoder} \rightarrow \mathcal{L}_{\text{cd}} \]

Lample et al. 2018b
The same model must be used for:
- L1→L1 and L2→L2 (denoising auto-encoding)
- L1→L2 and L2→L1 (translation)

Encoder representation should be language-agnostic

Lample et al. 2018b use adversarial training

Artetxe et al. 2018
• BPE-level model
  • Previously word-level models

• Single unified model
  • Single encoder, single decoder

• No adversarial training
Shared BPEs

- Take L1 and L2 monolingual corpora and split on BPE-level
  - Jointly learn on both languages to enable more frequent BPE sharing

- Train fasttext word embeddings

- Initialize NMT embeddings

- Embeddings are shown to be a useful initialization for UNMT

- Caveat: L1 and L2 need to share surface forms. Works well for English and German, not for English and Nepali.
Unified Model

- Single encoder and decoder (no discriminator or adversarial training)

- How do we deal with encoder language representation?

- Special language tag in decoder $<$2en$>$, $<$2de$>$

- Encoder has to learn a language-agnostic representation because the same representation can be used for denoising auto-encoding or translation

- Decoder is forced fed the language tag and knows what language to generate
Most of the basic techniques explained here are still used

Most works use pretrained LMs as initialization (next lecture)

UNMT does not work:
- for distant languages
- when the source and target domain are not comparable
- when there is insufficient monolingual corpora available for at least one of the languages
Summary

- **Transformer**
  - State-of-the-art NMT architecture

- **Document-level NMT**
  - Important for coherent translations across a document

- **Unsupervised NMT**
  - Important for enabling translation across a wide range of language pairs
Thank you for your attention


References IV


