Generative models of word representation

DGM4NLP

Miguel Rios
University of Amsterdam

April 28, 2019

Outline

1. Word embeddings
2. EmbedAlign
   - Model
   - Evaluation
   - Conclusions and Future Work
3. EmbedAlign-2

Introduction

Discriminative embedding models

word2vec

In the event of a chemical spill, most children know they should **evacuate** *as advised by people in charge.*

Place words in $\mathbb{R}^d$ as to answer questions like

"Have I seen this word in this context?"

Introduction

Discriminative embedding models

word2vec

In the event of a chemical spill, most children know they should **evacuate** *as advised by people in charge.*

Place words in $\mathbb{R}^d$ as to answer questions like

"Have I seen this word in this context?"

Fit a binary classifier

- positive examples
- negative examples

CoVe

- [McCann et al., 2017] Contextual Word Vectors are word embeddings learned by using the encoder from seq2seq with attention.

CoVe

- [McCann et al., 2017] Contextual Word Vectors are word embeddings learned by using the encoder from seq2seq with attention.
- NMT [Bahdanau et al., 2015] model is composed of a bi-LSTM encoder and an attentional LSTM decoder.

CoVe

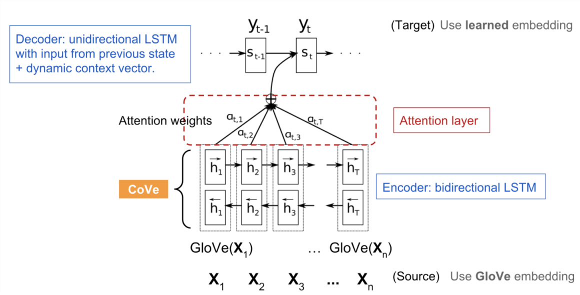
- [McCann et al., 2017] Contextual Word Vectors are word embeddings learned by using the encoder from seq2seq with attention.
- NMT [Bahdanau et al., 2015] model is composed of a bi-LSTM encoder and an attentional LSTM decoder.
- Trained on the English-German translation task.

CoVe

- [McCann et al., 2017] Contextual Word Vectors are word embeddings learned by using the encoder from seq2seq with attention.
- NMT [Bahdanau et al., 2015] model is composed of a bi-LSTM encoder and an attentional LSTM decoder.
- Trained on the English-German translation task.
- The encoder learns the embedding vectors of English words to translate them into German.

CoVe

- Motivation: The encoder captures high-level semantic and syntactic meanings.
  The encoder output is used on various downstream NLP tasks.



CoVe

- $\text{CoVe}(x) = \text{biLSTM}\left(\text{GloVe}\left(x\right)\right)$

CoVe

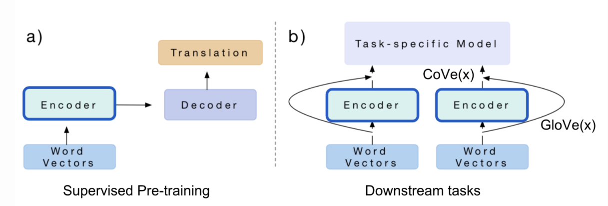
- $\text{CoVe}(x) = \text{ biLSTM }(\text{GloVe }(x))$
- Concatenation of GloVe and CoVe for question-answering.

CoVe

- $\text{CoVe}(x) = \text{ biLSTM } (\text{GloVe } (x))$
- Concatenation of GloVe and CoVe for question-answering.
- GloVe learns from word co-occurrences no sentence context,

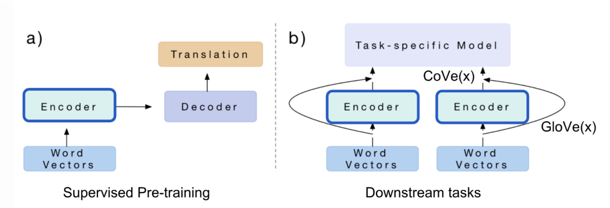
CoVe

- $\text{CoVe}(x) = \text{ biLSTM } (\text{GloVe } (x))$
- Concatenation of GloVe and CoVe for question-answering.
- GloVe learns from word co-occurrences no sentence context,
- CoVe is generated by processing text sequences is able to capture the contextual information.



CoVe

- $\text{CoVe}(x) = \text{ biLSTM } (\text{GloVe } (x))$
- Concatenation of GloVe and CoVe for question-answering.
- GloVe learns from word co-occurrences no sentence context,
- CoVe is generated by processing text sequences is able to capture the contextual information.



- Limitation use of parallel training data

ELMo

- Embeddings from Language Model [Peters et al., 2018] learns contextualized word embeddings with a language model.

ELMo

- Embeddings from Language Model [Peters et al., 2018] learns contextualized word embeddings with a language model.
- Bidirectional Language Model biLM.
  Input is a sequence of n words, $(x_1, \ldots, x_n)$, the language model learns to predict the probability of

ELMo

- Embeddings from Language Model [Peters et al., 2018] learns contextualized word embeddings with a language model.
- Bidirectional Language Model biLM.
  Input is a sequence of n words, $(x_1, \ldots, x_n)$, the language model learns to predict the probability of
- The forward contains words before the target:
  $p\left(x_1, \ldots, x_n\right) = \prod_{i=1}^{n} p\left(x_i | x_1, \ldots, x_{i-1}\right)$

ELMo

- Embeddings from Language Model [Peters et al., 2018] learns contextualized word embeddings with a language model.
- Bidirectional Language Model biLM.
  Input is a sequence of n words, $(x_1, \ldots, x_n)$, the language model learns to predict the probability of
- The forward contains words before the target:
  $p(x_1, \ldots, x_n) = \prod_{i=1}^{n} p(x_i | x_1, \ldots, x_{i-1})$
- The backward contains words after the target:
  $p(x_1, \ldots, x_n) = \prod_{i=1}^{n} p(x_i | x_{i+1}, \ldots, x_n)$

ELMo

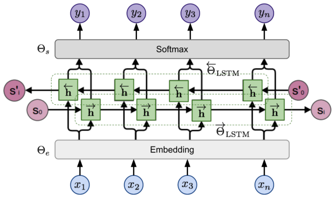
- Predictions from multi-layer LSTMs with hidden states $\overrightarrow{h}_{i,l}$ and $\overleftarrow{h}_{i,l}$ for input x_i.

ELMo

- Predictions from multi-layer LSTMs with hidden states $\overrightarrow{h}_{i,l}$ and $\overleftarrow{h}_{i,l}$ for input x_i.
- Final layer hidden state $h_{i,L} = [\overrightarrow{h}_{i,L}; \overleftarrow{h}_{i,L}]$ is used to output the probabilities over words.

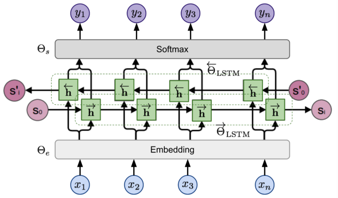
ELMo

- Predictions from multi-layer LSTMs with hidden states $\overrightarrow{h}_{i,l}$ and $\overleftarrow{h}_{i,l}$ for input x_i.
- Final layer hidden state $h_{i,L} = [\overrightarrow{h}_{i,L}; \overleftarrow{h}_{i,L}]$ is used to output the probabilities over words.
- Share the embedding layer and the softmax layer, parameterized by Θ_e and Θ_s respectively.



ELMo

- Predictions from multi-layer LSTMs with hidden states $\overrightarrow{h}_{i,l}$ and $\overleftarrow{h}_{i,l}$ for input x_i.
- Final layer hidden state $h_{i,L} = [\overrightarrow{h}_{i,L}; \overleftarrow{h}_{i,L}]$ is used to output the probabilities over words.
- Share the embedding layer and the softmax layer, parameterized by Θ_e and Θ_s respectively.



- Objective negative log likelihood in both directions:

$$\mathcal{L} = -\sum_{i=1}^{n} \left(\log p\left(x_i | x_1, \ldots, x_{i-1}; \Theta_e, \Theta_{\text{fwLSTM}}, \Theta_s\right) + \log p\left(x_i | x_{i+1}, \ldots, x_n; \Theta_e, \Theta_{\text{bwLSTM}}, \Theta_s\right)\right)$$

ELMo

- The top of a L-layer biLM, stacks all the hidden states across layers.
  The hidden state representation for x_i contains $2L+1$ vectors:

ELMo

- The top of a L-layer biLM, stacks all the hidden states across layers.
  The hidden state representation for x_i contains $2L+1$ vectors:
- $R_i = \{\mathbf{h}_{i,\ell} | \ell = 0, \ldots, L\}$ where $h_{0,l}$ is the embedding layer output and $h_{i,L} = [\overrightarrow{h}_{i,l}; \overleftarrow{h}_{i,l}]$.

ELMo

- The top of a L-layer biLM, stacks all the hidden states across layers.
  The hidden state representation for x_i contains $2L+1$ vectors:
- $R_i = \{\mathbf{h}_{i,\ell} | \ell = 0, \dots, L\}$ where $h_{0,l}$ is the embedding layer output and $h_{i,L} = [\overrightarrow{h}_{i,l}; \overleftarrow{h}_{i,l}]$.
- The weights, s^t task are learned for each end task The scaling factor γ^t is used to correct the misalignment between the distribution of biLM hidden states and the distribution of task specific representations.
  $v_i = f\left(R_i; \Theta^t\right) = \gamma^t \sum_{\ell=0}^{L} s_i^t \mathbf{h}_{i,\ell}$

ELMo

- To evaluate what information is captured by hidden states across different layers use on semantic and syntax tasks respectively with representations in different layers of biLM:

ELMo

- To evaluate what information is captured by hidden states across different layers use on semantic and syntax tasks respectively with representations in different layers of biLM:
- Semantic task: The word sense disambiguation (WSD) predict the meaning of a word given a context.
  The biLM top layer is better at this task than the first layer.

ELMo

- To evaluate what information is captured by hidden states across different layers use on semantic and syntax tasks respectively with representations in different layers of biLM:
- Semantic task: The word sense disambiguation (WSD) predict the meaning of a word given a context.
  The biLM top layer is better at this task than the first layer.
- Syntax task: The part-of-speech (POS) tagging task aims to infer the grammatical role of a word in one sentence.
  A higher accuracy can be achieved by using the biLM first layer than the top layer.

GPT

- Generative Pre-training Transformer [Radford, 2018] is a much larger LM.

GPT

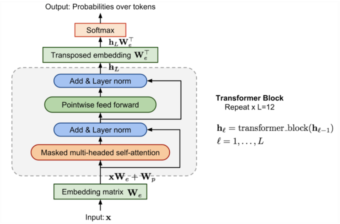
- Generative Pre-training Transformer [Radford, 2018] is a much larger LM.
- GPT is a multi-layer transformer decoder.

GPT

- Generative Pre-training Transformer [Radford, 2018] is a much larger LM.
- GPT is a multi-layer transformer decoder.
- GPT fine-tunes the same base model for all end tasks.

GPT

- Generative Pre-training Transformer [Radford, 2018] is a much larger LM.
- GPT is a multi-layer transformer decoder.
- GPT fine-tunes the same base model for all end tasks.
- Transformer Decoder:
  The model avoids the encoder part, only one single input sentence rather than source and target sequences.



GPT

- Each block contains a masked multi-headed self-attention [Vaswani et al., 2017] layer and a feed-forward layer.
  The final output produces a distribution over target words.

GPT

- Each block contains a masked multi-headed self-attention [Vaswani et al., 2017] layer and a feed-forward layer.
  The final output produces a distribution over target words.
- The loss is the log-likelihood LM, but without backward computation.
  $\mathcal{L}_{\mathrm{LM}} = -\sum_i \log p\left(x_i | x_{i-k}, \ldots, x_{i-1}\right)$

GPT

- Each block contains a masked multi-headed self-attention [Vaswani et al., 2017] layer and a feed-forward layer.
  The final output produces a distribution over target words.
- The loss is the log-likelihood LM, but without backward computation.
  $\mathcal{L}_{\mathrm{LM}} = -\sum_i \log p\left(x_i | x_{i-k}, \ldots, x_{i-1}\right)$
- Byte Pair Encoding (BPE) [Sennrich et al., 2015] over the input sequences. Motivation that rare and unknown words can be decomposed into multiple subwords. BPE finds the best word segmentation by iteratively and greedily merging frequent pairs of characters.

GPT

- Each block contains a masked multi-headed self-attention [Vaswani et al., 2017] layer and a feed-forward layer.
  The final output produces a distribution over target words.
- The loss is the log-likelihood LM, but without backward computation.
  $\mathcal{L}_{\text{LM}} = -\sum_i \log p\left(x_i | x_{i-k}, \ldots, x_{i-1}\right)$
- Byte Pair Encoding (BPE) [Sennrich et al., 2015] over the input sequences. Motivation that rare and unknown words can be decomposed into multiple subwords. BPE finds the best word segmentation by iteratively and greedily merging frequent pairs of characters.
- Supervised Fine-Tuning use the pre-trained language model directly

GPT

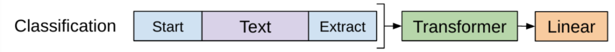
- For example in classification, each input has n tokens, $x = (x_1, \dots, x_n)$, and labels y.

GPT

- For example in classification, each input has n tokens, $x = (x_1, \ldots, x_n)$, and labels y.
- GPT processes the input sequence x by the pre-trained transformer decoder
  and the last layer output for x_n is $h_L^{(n)}$.

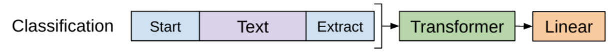
GPT

- For example in classification, each input has n tokens, $x = (x_1, \ldots, x_n)$, and labels y.
- GPT processes the input sequence x by the pre-trained transformer decoder
  and the last layer output for x_n is $h_L^{(n)}$.
- with weight matrix W_y to predict a distribution over class labels.



GPT

- For example in classification, each input has n tokens, $x = (x_1, \ldots, x_n)$, and labels y.
- GPT processes the input sequence x by the pre-trained transformer decoder
  and the last layer output for x_n is $h_L^{(n)}$.
- with weight matrix W_y to predict a distribution over class labels.



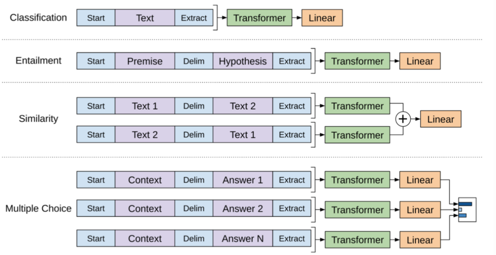
- $p(y|x_1, \ldots, x_n) = \mathrm{Cat}(\mathrm{softmax}\left(\mathbf{h}_L^{(n)}\mathbf{W}_y\right))$

GPT

- $$\begin{aligned}\mathcal{L}_{\text{cls}} &= \sum_{(\mathrm{x},y)\in\mathcal{D}} \log p\left(y|x_1,\ldots,x_n\right)\\ \mathcal{L}_{\text{LM}} &= -\sum_i \log p\left(x_i|x_{i-k},\ldots,x_{i-1}\right)\\ \mathcal{L} &= \mathcal{L}_{\text{cls}} + \lambda\mathcal{L}_{\text{LM}}\end{aligned} \tag{1}$$

GPT

- $$\begin{aligned}\mathcal{L}_{\text{cls}} &= \sum_{(\mathrm{x},y)\in\mathcal{D}} \log p\left(y|x_1,\ldots,x_n\right)\\ \mathcal{L}_{\text{LM}} &= -\sum_i \log p\left(x_i|x_{i-k},\ldots,x_{i-1}\right)\\ \mathcal{L} &= \mathcal{L}_{\text{cls}} + \lambda\mathcal{L}_{\text{LM}}\end{aligned} \tag{1}$$

- 

BERT

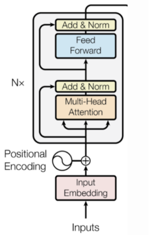
- Bidirectional Encoder Representations from Transformers [Devlin et al., 2018] trains a large language model, and then fine-tunes on specific tasks.

BERT

- Bidirectional Encoder Representations from Transformers [Devlin et al., 2018] trains a large language model, and then fine-tunes on specific tasks.
- The model architecture of BERT is a multi-layer bidirectional Transformer encoder.

BERT

- Bidirectional Encoder Representations from Transformers [Devlin et al., 2018] trains a large language model, and then fine-tunes on specific tasks.
- The model architecture of BERT is a multi-layer bidirectional Transformer encoder.
- BERT is trained with two auxiliary tasks instead of only the LM.



BERT

- Task1 : Mask language model (MLM)
  The cloze test consist in deleting a portion of certain items, words, or signs, where the participant replaces the missing item.

BERT

- Task1 : Mask language model (MLM)
  The cloze test consist in deleting a portion of certain items, words, or signs, where the participant replaces the missing item.
- Randomly mask 15% of tokens in each sequence with a special placeholder [MASK],

BERT

- Task1 : Mask language model (MLM)
  The cloze test consist in deleting a portion of certain items, words, or signs, where the participant replaces the missing item.
- Randomly mask 15% of tokens in each sequence with a special placeholder [MASK],
- BERT heuristics:

BERT

- Task1 : Mask language model (MLM)
  The cloze test consist in deleting a portion of certain items, words, or signs, where the participant replaces the missing item.
- Randomly mask 15% of tokens in each sequence with a special placeholder [MASK],
- BERT heuristics:
  - 80% probability, replace the chosen words with [MASK],

BERT

- Task1 : Mask language model (MLM)
  The cloze test consist in deleting a portion of certain items, words, or signs, where the participant replaces the missing item.
- Randomly mask 15% of tokens in each sequence with a special placeholder [MASK],
- BERT heuristics:
  - 80% probability, replace the chosen words with [MASK],
  - 10% probability, replace with a random word,

BERT

- Task1 : Mask language model (MLM)
  The cloze test consist in deleting a portion of certain items, words, or signs, where the participant replaces the missing item.
- Randomly mask 15% of tokens in each sequence with a special placeholder [MASK],
- BERT heuristics:
  - 80% probability, replace the chosen words with [MASK],
  - 10% probability, replace with a random word,
  - 10% probability, keep it the same.

BERT

- Task2: Next sentence prediction
  Motivation downstream tasks involve the understanding of relationships between sentences

BERT

- Task2: Next sentence prediction
  Motivation downstream tasks involve the understanding of relationships between sentences
- BERT adds another auxiliary task for training a classifier on whether one sentence is the next sentence of the other:

BERT

- Task2: Next sentence prediction
  Motivation downstream tasks involve the understanding of relationships between sentences
- BERT adds another auxiliary task for training a classifier on whether one sentence is the next sentence of the other:
- Sample sentence pairs (A, B) so that:

BERT

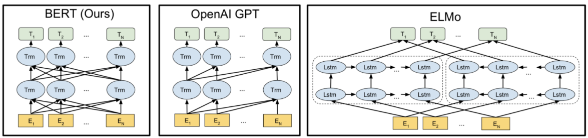
- Task2: Next sentence prediction
  Motivation downstream tasks involve the understanding of relationships between sentences
- BERT adds another auxiliary task for training a classifier on whether one sentence is the next sentence of the other:
- Sample sentence pairs (A, B) so that:
  - 50% of the time, B follows A;

BERT

- Task2: Next sentence prediction
  Motivation downstream tasks involve the understanding of relationships between sentences
- BERT adds another auxiliary task for training a classifier on whether one sentence is the next sentence of the other:
- Sample sentence pairs (A, B) so that:
  - 50% of the time, B follows A;
  - 50% of the time, B does not follow A.

BERT

- Task2: Next sentence prediction
  Motivation downstream tasks involve the understanding of relationships between sentences
- BERT adds another auxiliary task for training a classifier on whether one sentence is the next sentence of the other:
- Sample sentence pairs (A, B) so that:
  - 50% of the time, B follows A;
  - 50% of the time, B does not follow A.
- The model processes both sentences and output a binary label indicating whether B is the next sentence of A.



BERT

- The input embedding is the sum of three parts:

BERT

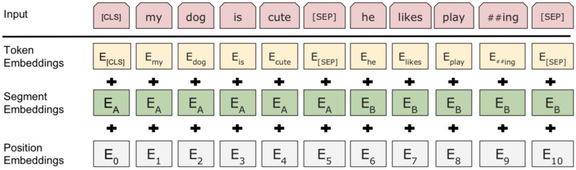
- The input embedding is the sum of three parts:
- WordPiece: words can be further divided into smaller sub-word units, it is more effective to handle rare or unknown words.

BERT

- The input embedding is the sum of three parts:
- WordPiece: words can be further divided into smaller sub-word units, it is more effective to handle rare or unknown words.
- Segment embeddings: sentence A embeddings and sentence B embeddings and separated by [SEP].

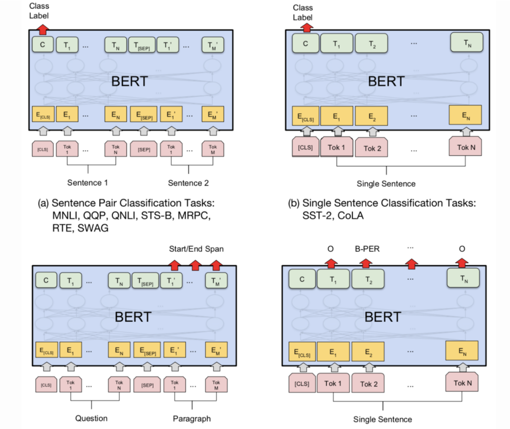
BERT

- The input embedding is the sum of three parts:
- WordPiece: words can be further divided into smaller sub-word units, it is more effective to handle rare or unknown words.
- Segment embeddings: sentence A embeddings and sentence B embeddings and separated by [SEP].
- Position embeddings: Positional embeddings are learned instead of hard-coded.



BERT

- BERT fine-tuning requires the final hidden state of the special first token [CLS], $h_L^{[CLS]}$.



GPT-2

- GPT-2 has 1.5B parameters, 10x more than the original GP and it achieves SOTA results on 7 out of 8 NLP in a zero-shot transfer setting without any task-specific fine-tuning.

GPT-2

- GPT-2 has 1.5B parameters, 10x more than the original GP and it achieves SOTA results on 7 out of 8 NLP in a zero-shot transfer setting without any task-specific fine-tuning.
- The pre-training dataset contains 8 million Web pages collected by crawling qualified outbound links from Reddit.

GPT-2

- GPT-2 has 1.5B parameters, 10x more than the original GP and it achieves SOTA results on 7 out of 8 NLP in a zero-shot transfer setting without any task-specific fine-tuning.
- The pre-training dataset contains 8 million Web pages collected by crawling qualified outbound links from Reddit.
- Large improvements by GPT-2 are specially noticeable on small datasets and datasets used for measuring long-term dependency.

GPT-2

- GPT-2 has 1.5B parameters, 10x more than the original GP and it achieves SOTA results on 7 out of 8 NLP in a zero-shot transfer setting without any task-specific fine-tuning.
- The pre-training dataset contains 8 million Web pages collected by crawling qualified outbound links from Reddit.
- Large improvements by GPT-2 are specially noticeable on small datasets and datasets used for measuring long-term dependency.
- Zero-Shot Transfer: All the downstream language tasks are framed as predicting conditional probabilities and there is no task-specific fine-tuning.

Outline

1. Word embeddings
2. EmbedAlign
   - Model
   - Evaluation
   - Conclusions and Future Work
3. EmbedAlign-2

VAE Recap

- As a NN the VAE consists of an encoder, a decoder, and a loss function.

VAE Recap

- As a NN the VAE consists of an encoder, a decoder, and a loss function.
- The encoder is a NN with input x, and output latent representation z.

VAE Recap

- As a NN the VAE consists of an encoder, a decoder, and a loss function.
- The encoder is a NN with input x, and output latent representation z.
- The lower-dimensional space is stochastic and parametrise $q_\phi(z \mid x)$ which is a Gaussian probability density.

VAE Recap

- As a NN the VAE consists of an encoder, a decoder, and a loss function.
- The encoder is a NN with input x, and output latent representation z.
- The lower-dimensional space is stochastic and parametrise $q_\phi(z \mid x)$ which is a Gaussian probability density.
- The decoder is a NN with input z, and parametrize the probability distribution of the data.
  The decoder is denoted by $p_\theta(x \mid z)$.

VAE Recap

- As a NN the VAE consists of an encoder, a decoder, and a loss function.
- The encoder is a NN with input x, and output latent representation z.
- The lower-dimensional space is stochastic and parametrise $q_\phi(z \mid x)$ which is a Gaussian probability density.
- The decoder is a NN with input z, and parametrize the probability distribution of the data.
  The decoder is denoted by $p_\theta(x \mid z)$.
- The loss is the negative log-likelihood with a regulariser.
  $\mathcal{L}(\theta, \phi|x) = \mathbb{E}_{q(z|x)}\left[\log p_\theta\left(x|z\right)\right] - \mathrm{KL}\left(q_\phi(z|x)\|p_\theta(z)\right)$

The reparametrisation trick

- How to take derivatives with respect to the parameters of a stochastic variable.

The reparametrisation trick

- How to take derivatives with respect to the parameters of a stochastic variable.
- Given z sample from a distribution $q_\phi(z \mid x)$

The reparametrisation trick

- How to take derivatives with respect to the parameters of a stochastic variable.
- Given z sample from a distribution $q_\phi(z \mid x)$
- The z sample is fixed, but the derivative should be nonzero.

The reparametrisation trick

- How to take derivatives with respect to the parameters of a stochastic variable.
- Given z sample from a distribution $q_\phi(z \mid x)$
- The z sample is fixed, but the derivative should be nonzero.
- It is possible to reparametrise samples, for example, in a normally-distributed variable with mean μ and standard deviation σ,

The reparametrisation trick

- How to take derivatives with respect to the parameters of a stochastic variable.
- Given z sample from a distribution $q_\phi(z \mid x)$
- The z sample is fixed, but the derivative should be nonzero.
- It is possible to reparametrise samples, for example, in a normally-distributed variable with mean μ and standard deviation σ,
- we can sample from it like this:
  $z = \mu + \sigma \odot \epsilon$
  where $\epsilon \sim Normal(0, I)$.

Discriminative embedding models

In the event of a chemical spill, most children know they should **evacuate** *as advised by people in charge.*

- Limitations

Discriminative embedding models

In the event of a chemical spill, most children know they should **evacuate** *as advised by people in charge.*

- Limitations
  - Representation learning is an unsupervised problem we only observe positive/complete context

Discriminative embedding models

In the event of a chemical spill, most children know they should **evacuate** *as advised by people in charge.*

- Limitations
  - Representation learning is an unsupervised problem we only observe positive/complete context
  - Distributional hypothesis is strong but fails when context is not discriminative

Discriminative embedding models

In the event of a chemical spill, most children know they should **evacuate** *as advised by people in charge.*

- Limitations
  - Representation learning is an unsupervised problem we only observe positive/complete context
  - Distributional hypothesis is strong but fails when context is not discriminative
  - Word senses are collapsed into one vector

Embedalign

- Generative model to induce word representations

Embedalign

In the event of a chemical spill, most children know they should **evacuate** *as advised by people in charge.*

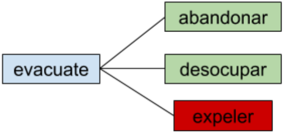
- Generative model to induce word representations
- Learn from positive examples

Embedalign

In the event of a chemical spill, most children know they should **evacuate** *as advised by people in charge.*

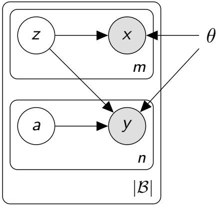
- Generative model to induce word representations
- Learn from positive examples
- Learn from richer (less ambiguous) context
  Foreign text is proxy to sense supervision (Diab and Resnik, 2002)

En caso de un derrame de productos químicos, la mayoría de los niños saben que deben **abandonar** *el lugar según lo aconsejado por las personas a cargo.*



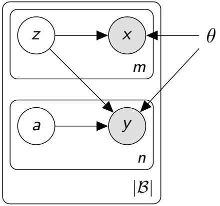
Generative Model

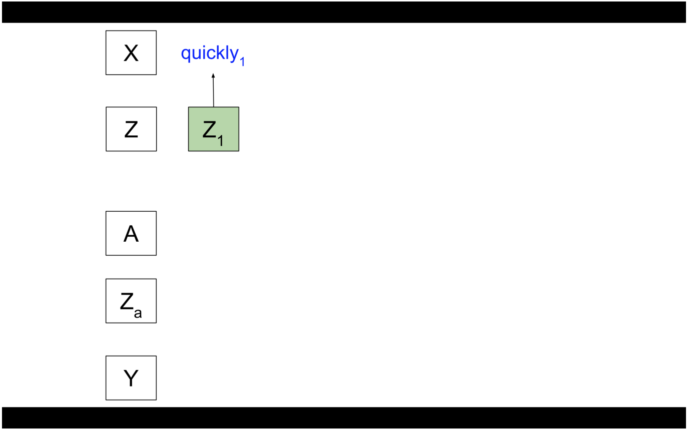
quickly evacuate the area / deje el lugar rápidamente



Generative Model

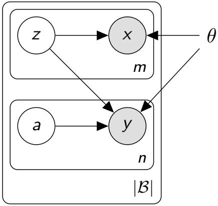
quickly evacuate the area / deje el lugar rápidamente

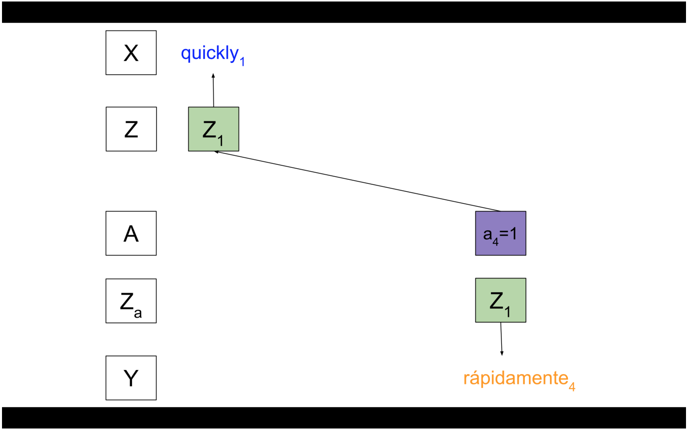




Generative Model

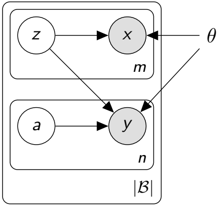
quickly evacuate the area / deje el lugar rápidamente

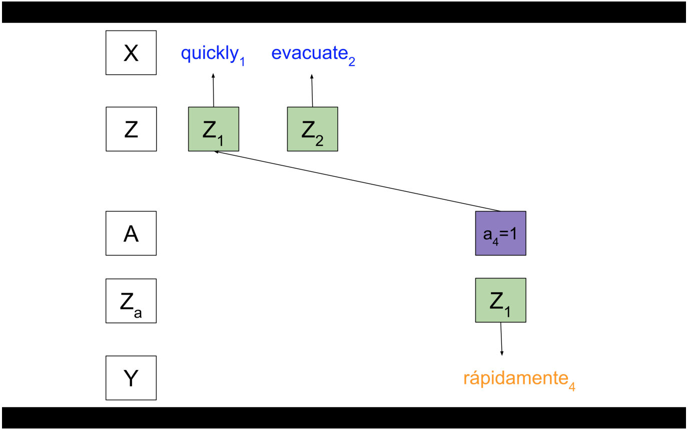




Generative Model

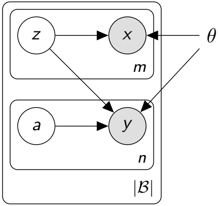
quickly evacuate the area / deje el lugar rápidamente

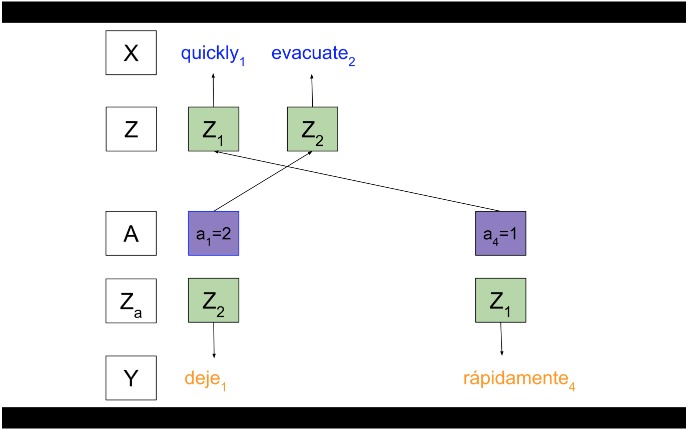




Generative Model

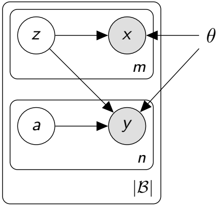
quickly evacuate the area / deje el lugar rápidamente

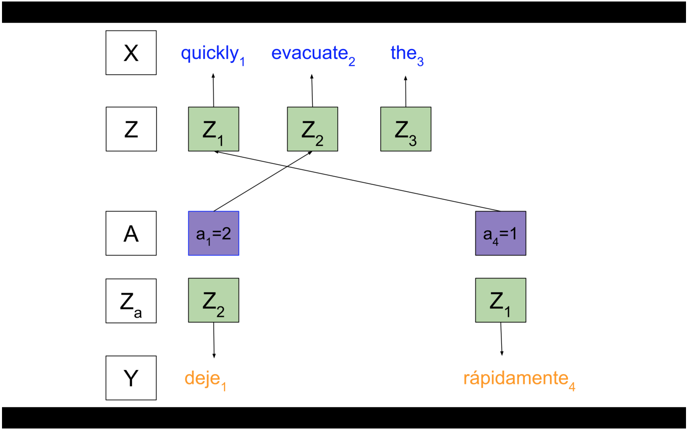




Generative Model

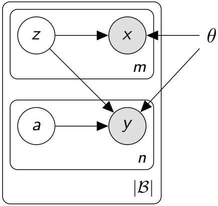
quickly evacuate the area / deje el lugar rápidamente

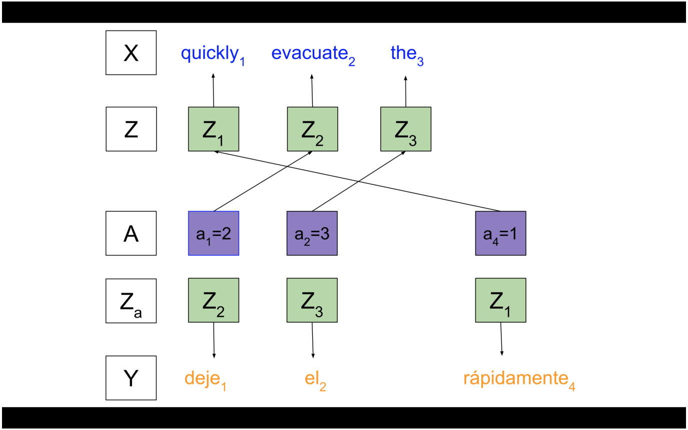




Generative Model

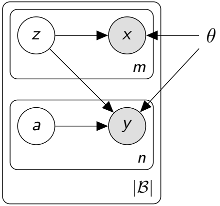
quickly evacuate the area / deje el lugar rápidamente

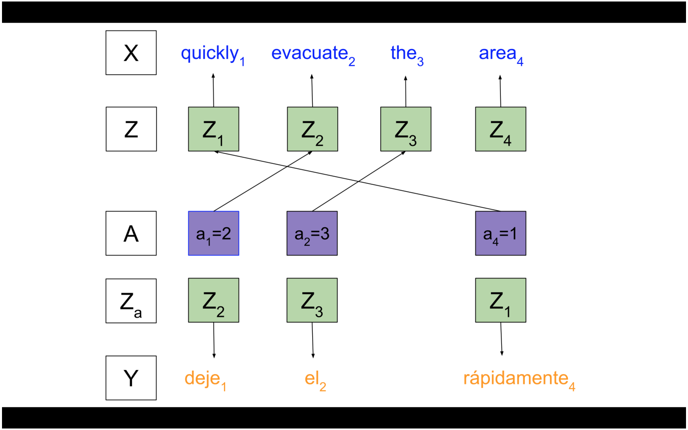




Generative Model

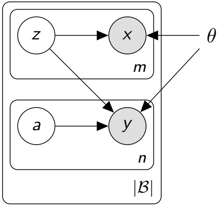
quickly evacuate the area / deje el lugar rápidamente

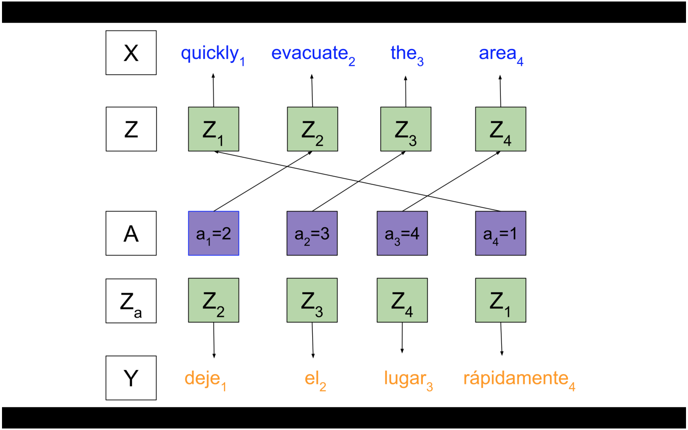




Generative Model

quickly evacuate the area / deje el lugar rápidamente



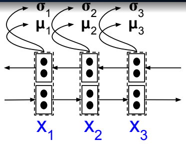


Learning

1. Read sentence

Learning

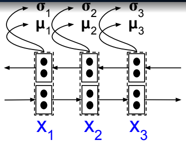
1. Read sentence
2. Predict posterior mean μ_i and std σ_i



evacuate$_1$ the$_2$ area$_3$

Learning

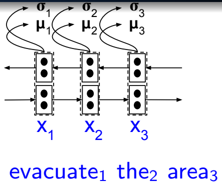
1. Read sentence
2. Predict posterior mean μ_i and std σ_i
3. Sample $z_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$



evacuate$_1$ the$_2$ area$_3$

Learning

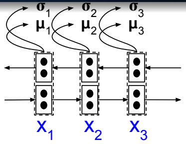
1. Read sentence
2. Predict posterior mean μ_i and std σ_i
3. Sample $z_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$



4. Predict categorical distributions

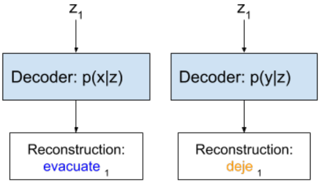
Learning

1. Read sentence
2. Predict posterior mean μ_i and std σ_i
3. Sample $z_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$



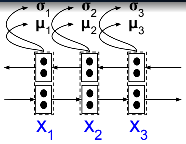
evacuate$_1$ the$_2$ area$_3$

4. Predict categorical distributions
5. Generate observations
   evacuate$_1$ the$_2$ area$_3$ / deje$_1$ el$_2$ lugar$_3$



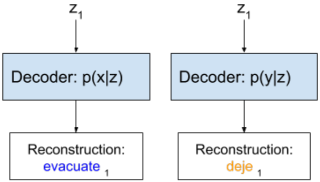
Learning

1. Read sentence
2. Predict posterior mean μ_i and std σ_i
3. Sample $z_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$



evacuate$_1$ the$_2$ area$_3$

4. Predict categorical distributions
5. Generate observations
   evacuate$_1$ the$_2$ area$_3$ / deje$_1$ el$_2$ lugar$_3$



6. Maximise a lowerbound on likelihood (Kingma and Welling, 2014)

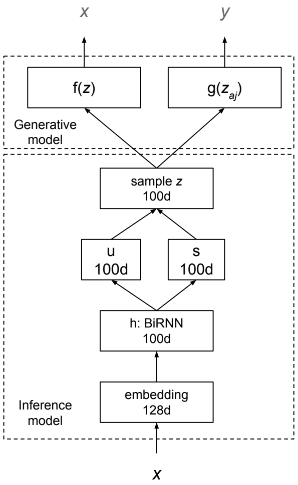
log p(x|z) + log p(y|z) − KL(q || p)

Marginalise **a**

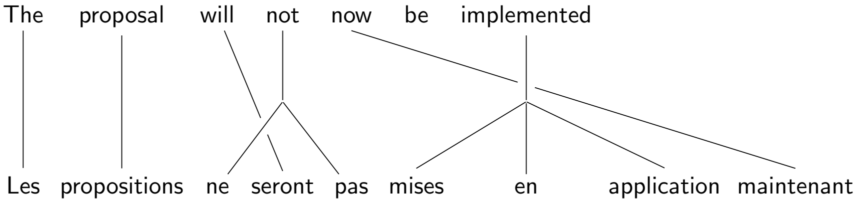
Data

Corpus	Sentence pairs (million)	Tokens (million)
Europarl EN-FR	1.7	42.5
Europarl EN-DE	1.7	43.5

Architecture



Word Alignment



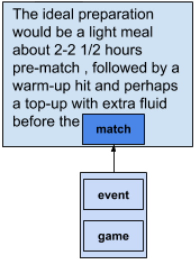
Word Alignment

- Model selection on Dev set

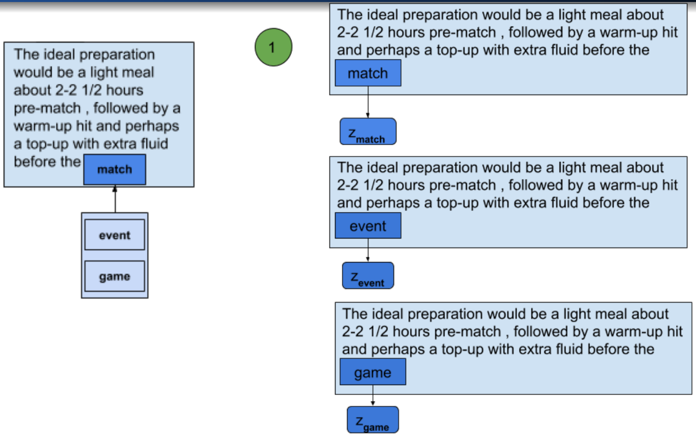
AER $\downarrow$

Model	En-Fr	En-De
IBM1	32.45	46.71
IBM2	22.61	40.11
EmbAlign	29.43 ± 1.84	48.09 ± 2.12

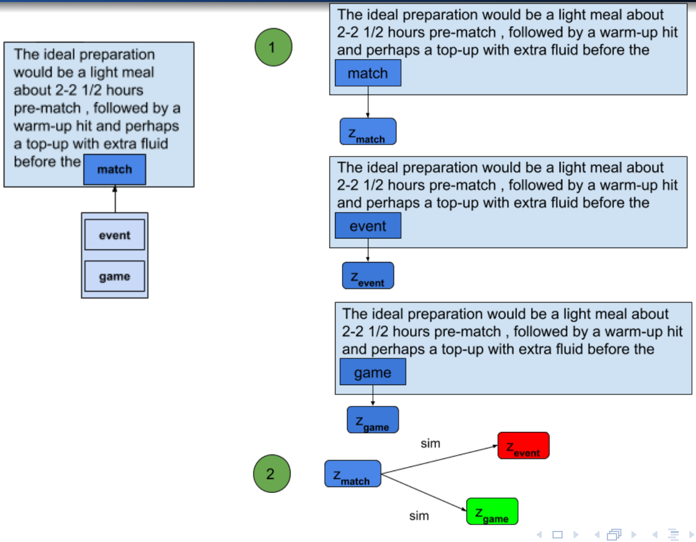
Lexical Substitution



Lexical Substitution



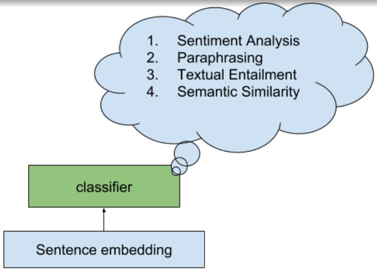
Lexical Substitution



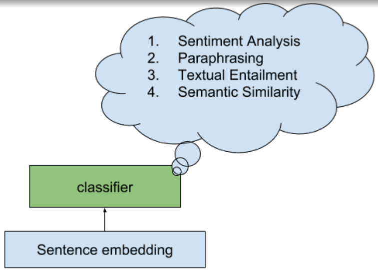
Lexical Substitution

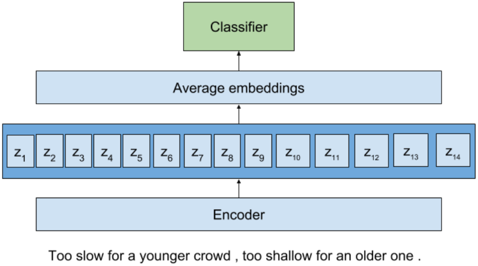
Model	GAP ↑	Training size
RANDOM	30.0	
SKIPGRAM (Melamud et al., 2015)	44.9	ukWaC-2B
BSG (Bražinskas et al., 2017)	46.1	ukWaC-2B
EN	21.31 ± 1.05	
EN-FR	42.19 ± 0.57	Euro-42M
EN-DE	42.07 ± 0.47	

Sentence Evaluation (SentEval)

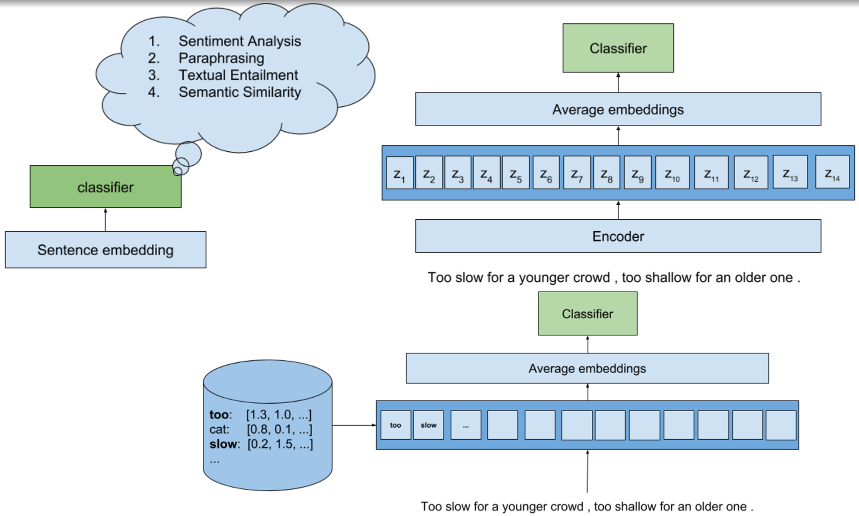


Sentence Evaluation (SentEval)





Sentence Evaluation (SentEval)



Sentence Evaluation (SentEval)

						ACC ↑	ACC/F1 ↑	CORR ↑		CORR ↑
Model	MR	CR	SUBJ	MPQA	SST	TREC	MRPC	SICK-R	SICK-E	STS14
SkipGram$_{\text{En}}$	**70.96**	**76.16**	**87.24**	**86.87**	**73.64**	65.20	70.7/80.1	0.710	76.2	0.45/0.49
En	57.5	67.1	72.0	70.8	57.0	58.0	70.6/80.3	0.648	74.4	0.59/0.59
En-Fr	64.0	71.8	79.1	81.5	64.7	58.4	72.1/81.2	0.682	74.6	0.60/0.59
En-De	62.6	68.0	77.3	82.0	65.0	66.8	70.4/79.8	0.681	75.5	0.58/0.58
Combo	66.1	72.4	82.4	84.4	69.8	**69.0**	**71.9**/**80.6**	**0.727**	**76.3**	**0.62**/**0.61**

Sentence Evaluation (SentEval)

						ACC ↑	ACC/F1 ↑	CORR ↑		CORR ↑
Model	MR	CR	SUBJ	MPQA	SST	TREC	MRPC	SICK-R	SICK-E	STS14
SkipGram$_{\text{En}}$	**70.96**	**76.16**	**87.24**	**86.87**	**73.64**	65.20	70.7/80.1	0.710	76.2	0.45/0.49
En	57.5	67.1	72.0	70.8	57.0	58.0	70.6/80.3	0.648	74.4	0.59/0.59
En-Fr	64.0	71.8	79.1	81.5	64.7	58.4	72.1/81.2	0.682	74.6	0.60/0.59
En-De	62.6	68.0	77.3	82.0	65.0	66.8	70.4/79.8	0.681	75.5	0.58/0.58
Combo	66.1	72.4	82.4	84.4	69.8	**69.0**	**71.9/80.6**	**0.727**	**76.3**	**0.62/0.61**

SkipGram (Conneau et al., 2017)	**77.7**	**79.8**	**90.9**	**88.3**	**79.7**	**83.6**	**72.5/81.4**	**0.803**	**78.7**	**0.65/0.64**
NMT$_{\text{En-Fr}}$ (Conneau et al., 2017)	64.7	70.1	84.8	81.5	-	82.8	-	-	-	0.42/0.43

Conclusions

- Generative training

Conclusions

- Generative training
  - model learns form positive examples

Conclusions

- Generative training
  - model learns form positive examples
  - no need for context window

Conclusions

- Generative training
  - model learns form positive examples
  - no need for context window
- Translation data

Conclusions

- Generative training
  - model learns form positive examples
  - no need for context window
- Translation data
  - less ambiguous embeddings

Conclusions

- Generative training
  - model learns form positive examples
  - no need for context window
- Translation data
  - less ambiguous embeddings
  - model helps with semantic tasks e.g. paraphrasing

Open Source Code

- Try pre-trained Europarl model on SentEval:
  `https://github.com/uva-slpl/embedalign/blob/master/notebooks/senteval_embedalign.ipynb`

Outline

1. Word embeddings
2. EmbedAlign
   - Model
   - Evaluation
   - Conclusions and Future Work
3. EmbedAlign-2

EmbedAlign-2

- Hierarchical generative model of words and sentences by exploiting automatically generated paraphrasing data.

EmbedAlign-2

- Hierarchical generative model of words and sentences by exploiting automatically generated paraphrasing data.
- Model embeds sentences and words as probability densities.

EmbedAlign-2

- Hierarchical generative model of words and sentences by exploiting automatically generated paraphrasing data.
- Model embeds sentences and words as probability densities.
- Model learns from sentence-aligned corpora by marginalisation of latent lexical alignments.

EmbedAlign-2

- Hierarchical generative model of words and sentences by exploiting automatically generated paraphrasing data.
- Model embeds sentences and words as probability densities.
- Model learns from sentence-aligned corpora by marginalisation of latent lexical alignments.
- Testing on different sentence representation benchmarks.

EmbedAlign-2

- Hierarchical generative model of words and sentences by exploiting automatically generated paraphrasing data.

EmbedAlign-2

- Hierarchical generative model of words and sentences by exploiting automatically generated paraphrasing data.
- Model embeds sentences and words as probability densities.

EmbedAlign-2

- Hierarchical generative model of words and sentences by exploiting automatically generated paraphrasing data.
- Model embeds sentences and words as probability densities.
- Model learns from sentence-aligned corpora by marginalisation of latent lexical alignments.

EmbedAlign-2

- Hierarchical generative model of words and sentences by exploiting automatically generated paraphrasing data.
- Model embeds sentences and words as probability densities.
- Model learns from sentence-aligned corpora by marginalisation of latent lexical alignments.
- Testing on different sentence representation benchmarks.

EmbedAlign-2

- Wieting and Gimpel (2017) port the pivoting approach in (Bannard and Callison-Burch, 2005) for paraphrasing to NMT. *Paraphrase English sentences by translating to a foreign language and back to English*.

EmbedAlign-2

- Wieting and Gimpel (2017) port the pivoting approach in (Bannard and Callison-Burch, 2005) for paraphrasing to NMT. *Paraphrase English sentences by translating to a foreign language and back to English*.
- How to represent paraphrases that introduce **syntactic variations**.

EmbedAlign-2

- Wieting and Gimpel (2017) port the pivoting approach in (Bannard and Callison-Burch, 2005) for paraphrasing to NMT. *Paraphrase English sentences by translating to a foreign language and back to English*.
- How to represent paraphrases that introduce **syntactic variations**.
- How to represent **noisy** automatically generated data.

Reference Translation	Machine Translation
so, what's half an hour?	half an hour won't kill you.
well, don't worry. i've taken out tons and tons of guys. lots of guys.	don't worry, i've done it to dozens of men.
it's gonna be ...... classic.	yeah, sure. it's gonna be great.
greetings, all!	hello everyone!
but she doesn't have much of a case.	but as far as the case goes, she doesn't have much.
it was good in spite of the taste.	despite the flavor, it felt good.

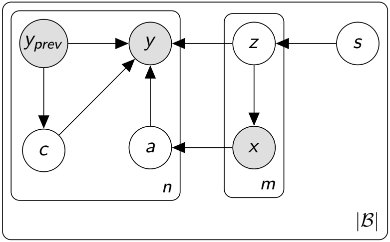
EmbedAlign-2

- Parallel data $\langle x_1^m, y_1^n \rangle$ where x_1^m is a sentence in original English and y_1^n is a sentence paraphrase English.

EmbedAlign-2

- Parallel data $\langle x_1^m, y_1^n \rangle$ where x_1^m is a sentence in original English and y_1^n is a sentence paraphrase English.
- Hierarchical generative model parameterised by neural networks.

$$
\begin{aligned}
S &\sim \mathcal{N}(0, I) \\
Z_i &\sim \mathcal{N}(\mu(s), \sigma^2(s))
\end{aligned}
\tag{2}
$$



Model I

- With a Gaussian distributed sentence embedding s and word embeddings z_1^m, we can marginalise collocation and alignment components.
  Latent decision is modelled with a Bernoulli trial:

$$p(y_j|x_1^m, z_1^m) = p(c_j = 0)p(y_j|y_{j-1}) + p(c_j = 1)\sum_{a_j=1}^{m} p(a_j|m, n)p(y_j|z_{a_j}) \quad (3)$$

Model II

- Inference:

$$q(s, z_1^m) = q_s(s) \prod_{i=1}^{m} q_{z_i}(z_i) \tag{4}$$

where $S \sim \mathcal{N}(\mathbf{u}_0, \mathbf{s}_0^2)$ and $Z_i \sim \mathcal{N}(\mathbf{u}_i, \mathbf{s}_i^2)$.

- Collocation is rudimentary language model condition on previously generated word.
- Alignment identifying relationships among words in the parallel data.

Model III

- Estimate parameters via maximisation of a lower-bound on marginal likelihood:

$$
\begin{aligned}
\log p(x_1^m, y_1^n) \geq \mathbb{E}[\log p(x_1^m|z_1^m)] \\
+\mathbb{E}[\log p(y_j|x_1^m, z_1^m)] \\
-\sum_{i=1}^{m} \mathbb{E}[\mathrm{KL}(q(z_i|x)\|p(z|s))] \\
-\mathrm{KL}(q(s|x)\|\mathcal{N}(0, I))
\end{aligned}
\tag{5}
$$

Evaluation

- Sentence-level paraphrases PARANMT-5M with 5 million sentences **EN1-EN2**.
- English-French **EN-FR** and English-German **EN-DE** from Europarl-v7.

Word Alignment

- Test alignment error rate (AER) on bilingual models.
- report on IBM models 1, 2 and FastAlign.

Model	En-Fr ↓	En-De ↓
IBM1	0.32	0.47
IBM2	0.23	0.40
EmbedAlign	0.29 ± 0.02	0.48 ± 0.02
FastAlign	0.19	0.36
This work	0.18 ± 0.01	0.40 ± 0.01

Evaluation

Model	MR	CR	MPQA	SUBJ	SST	TREC	MRPC	SICK-E	SICK-R	STS14
Baselines										
ELMO	79.87	84.85	89.21	94.19	85.67	92.80	72.93/80.90	81.21	0.82/0.75	0.61/0.58
SkipGram$_{\text{En1}}$	72.11	78.20	85.51	88.82	75.56	72.20	71.54/81.02	76.16	0.75/0.66	0.44/0.48
Ours										
En-Fr	66.76	71.18	85.40	82.32	67.52	70.45	71.90/80.77	75.18	0.67/0.62	0.49/0.50
En-De	66.00	72.21	85.71	81.63	67.64	70.45	71.83/80.85	75.73	0.66/0.62	0.49/0.59
En1-En2	66.88	71.59	81.80	82.97	69.14	67.30	71.33/80.36	75.62	0.72/0.66	0.53/0.52

EmbedAlign-2

- We modify alignment distribution

EmbedAlign-2

- We modify alignment distribution
  - From IBM1 to IBM2
    En-Fr 29.43 $\rightarrow$ 18.20 AER

EmbedAlign-2

- We modify alignment distribution
  - From IBM1 to IBM2
    En-Fr 29.43 $\rightarrow$ 18.20 AER
- We model word and sentence embeddings

EmbedAlign-2

- We modify alignment distribution
  - From IBM1 to IBM2
    En-Fr 29.43 $\rightarrow$ 18.20 AER
- We model word and sentence embeddings
  - **M**ovie **R**eviews 66.10 $\rightarrow$ 70.55 ACC

EmbedAlign-2

- We modify alignment distribution
  - From IBM1 to IBM2
    En-Fr 29.43 $\rightarrow$ 18.20 AER
- We model word and sentence embeddings
  - **M**ovie **R**eviews 66.10 $\rightarrow$ 70.55 ACC
  - **M**icrosoft **P**araphrase 71.90/80.6 $\rightarrow$ 72.93/81.27 ACC/F1

EmbedAlign-2

- We modify alignment distribution
  - From IBM1 to IBM2
    En-Fr 29.43 $\rightarrow$ 18.20 AER
- We model word and sentence embeddings
  - **M**ovie **R**eviews 66.10 $\rightarrow$ 70.55 ACC
  - **M**icrosoft **P**araphrase 71.90/80.6 $\rightarrow$ 72.93/81.27 ACC/F1
  - **S**ick **R** 0.727 $\rightarrow$ 0.770 CORR

EmbedAlign-2

- We modify alignment distribution
  - From IBM1 to IBM2
    En-Fr 29.43 → 18.20 AER
- We model word and sentence embeddings
  - **M**ovie **R**eviews 66.10 → 70.55 ACC
  - **M**icrosoft **P**araphrase 71.90/80.6 → 72.93/81.27 ACC/F1
  - **S**ick **R** 0.727 → 0.770 CORR
- We **will** expand the distributional context to multiple foreign languages at once

References I

TOREAD: Arthur Brazinskas, Serhii Havrylov, and Ivan Titov. Embedding words as distributions with a bayesian skip-gram model. arXiv preprint arXiv:1711.11027, 2017.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1409.0473`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

References II

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in translation: Contextualized word vectors. *CoRR*, abs/1708.00107, 2017. URL http://arxiv.org/abs/1708.00107.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *CoRR*, abs/1802.05365, 2018. URL http://arxiv.org/abs/1802.05365.

Alec Radford. Improving language understanding by generative pre-training. 2018.

References III

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909, 2015. URL http://arxiv.org/abs/1508.07909.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.