Deep Generative Language Models

DGM4NLP

Miguel Rios
University of Amsterdam

May 1, 2019

Outline

Recap Generative Models of Word Representation

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?"

Recap Generative Models of Word Representation

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

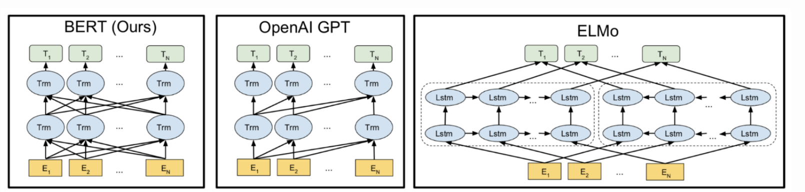
"Have I seen this word in this context?"

Fit a binary classifier

- positive examples
- negative examples

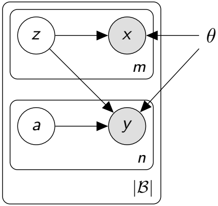
Recap Generative Models of Word Representation

- The models processes a sentence and outputs a word representation:



Recap Generative Models of Word Representation

quickly evacuate the area / deje el lugar rápidamente



[redacted]

X

Z

A

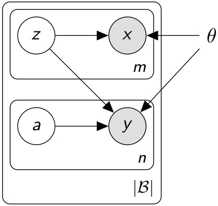
Z_a

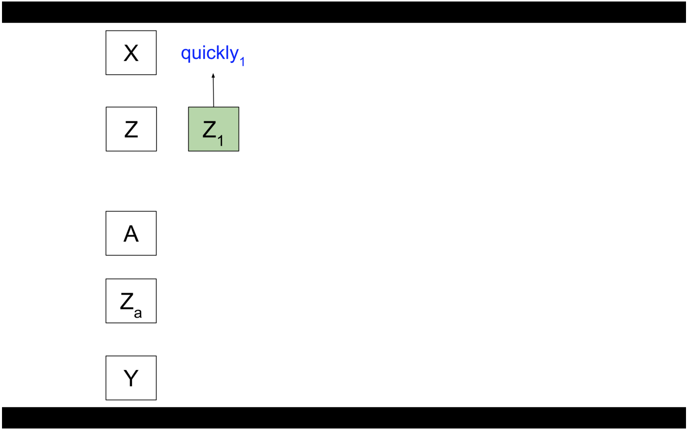
Y

[redacted]

Recap Generative Models of Word Representation

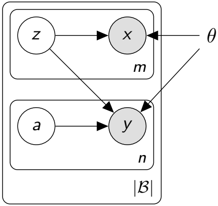
quickly evacuate the area / deje el lugar rápidamente

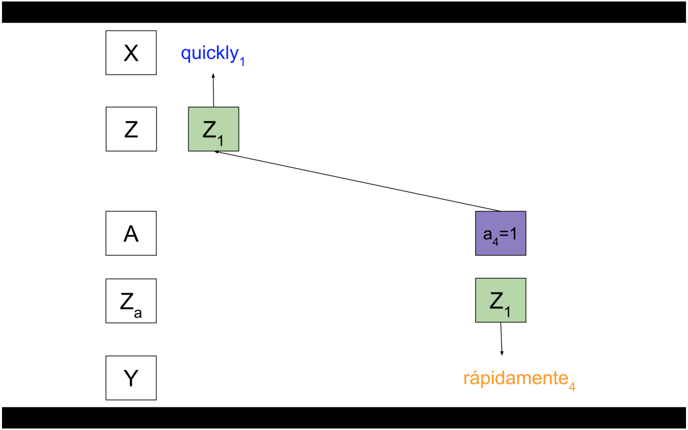




Recap Generative Models of Word Representation

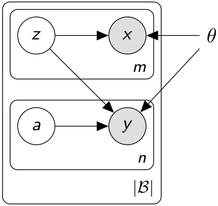
quickly evacuate the area / deje el lugar rápidamente

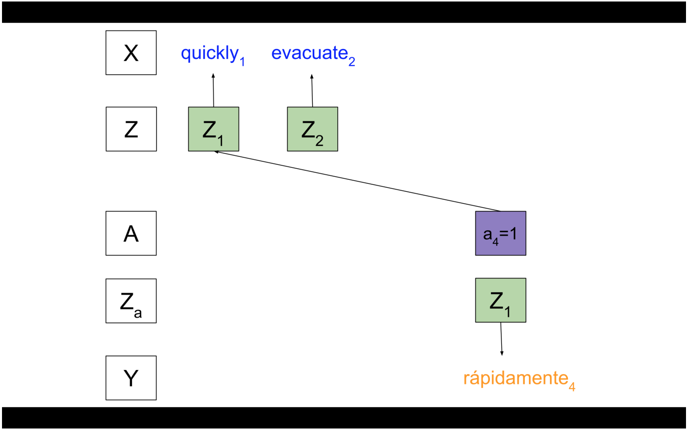




Recap Generative Models of Word Representation

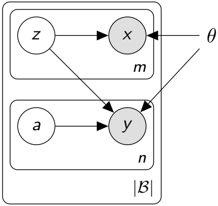
quickly evacuate the area / deje el lugar rápidamente

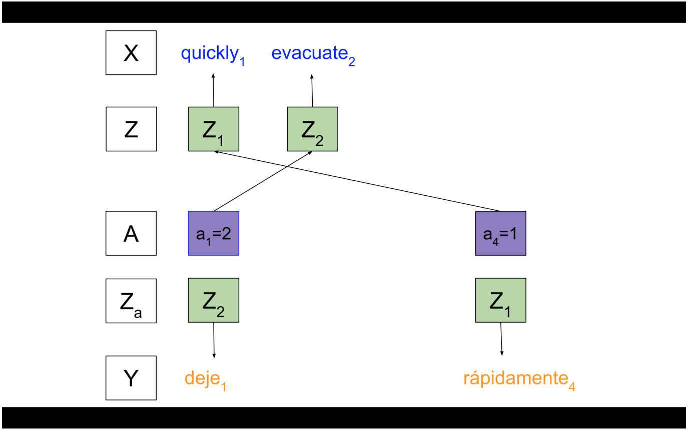




Recap Generative Models of Word Representation

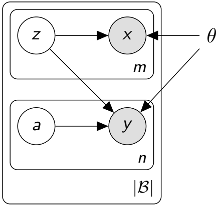
quickly evacuate the area / deje el lugar rápidamente

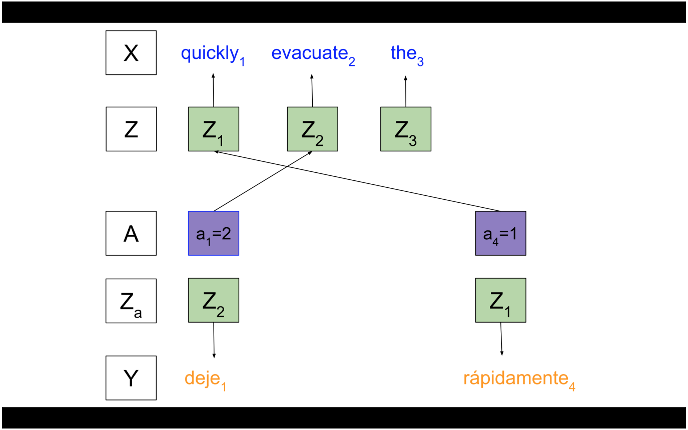




Recap Generative Models of Word Representation

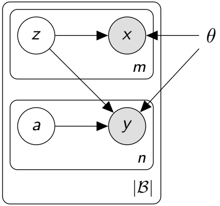
quickly evacuate the area / deje el lugar rápidamente

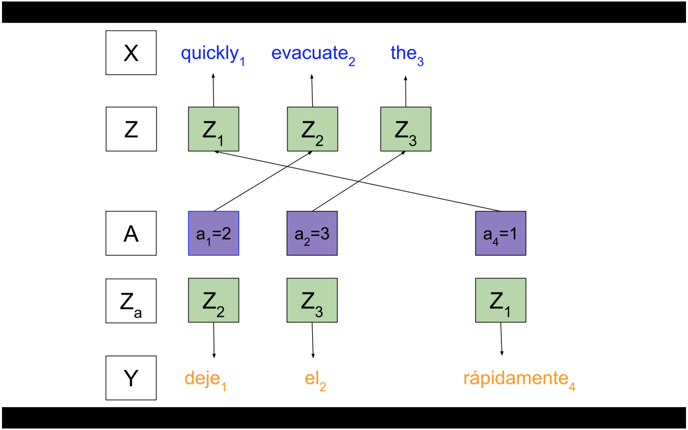




Recap Generative Models of Word Representation

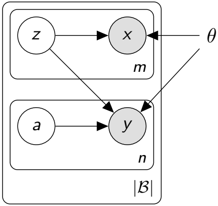
quickly evacuate the area / deje el lugar rápidamente

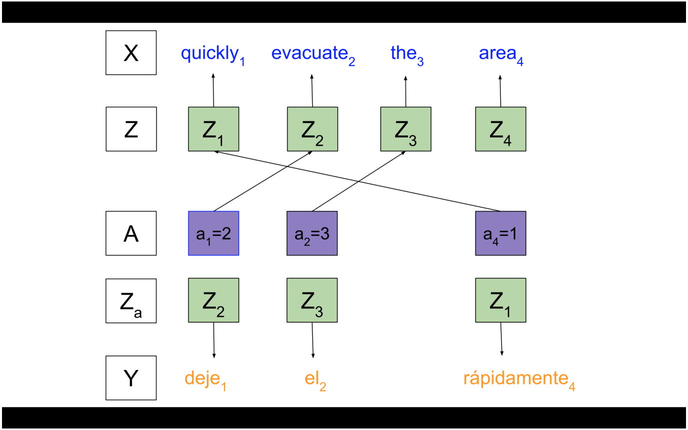




Recap Generative Models of Word Representation

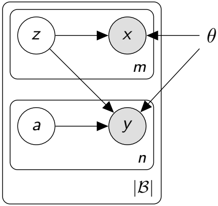
quickly evacuate the area / deje el lugar rápidamente

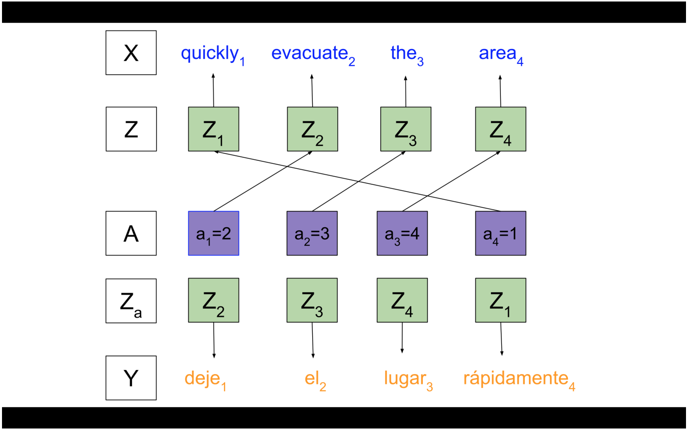




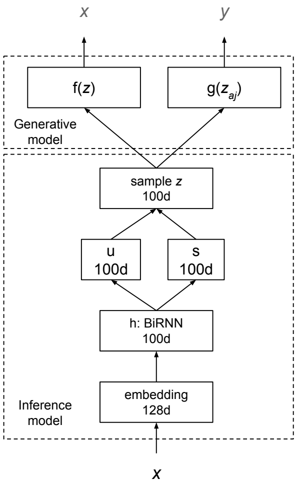
Recap Generative Models of Word Representation

quickly evacuate the area / deje el lugar rápidamente





Recap Generative Models of Word Representation



Introduction

- you know nothing, jon x

Introduction

- you know nothing, jon x
- ground control to major x

Introduction

- you know nothing, jon x
- ground control to major x
- the x

Introduction

- the quick brown x

Introduction

- the quick brown x
- the quick brown fox x

Introduction

- the quick brown x
- the quick brown fox x
- the quick brown fox jumps x

Introduction

- the quick brown x
- the quick brown fox x
- the quick brown fox jumps x
- the quick brown fox jumps over x

Introduction

- the quick brown x
- the quick brown fox x
- the quick brown fox jumps x
- the quick brown fox jumps over x
- the quick brown fox jumps over the x

Introduction

- the quick brown x
- the quick brown fox x
- the quick brown fox jumps x
- the quick brown fox jumps over x
- the quick brown fox jumps over the x
- the quick brown fox jumps over the lazy x

Introduction

- the quick brown x
- the quick brown fox x
- the quick brown fox jumps x
- the quick brown fox jumps over x
- the quick brown fox jumps over the x
- the quick brown fox jumps over the lazy x
- the quick brown fox jumps over the lazy dog

Definition

- Language models give us the probability of a sentence;

Definition

- Language models give us the probability of a sentence;
- At a time step, they assign a probability to the next word.

Applications

- Very useful on different tasks:

Applications

- Very useful on different tasks:
- Speech recognition;

Applications

- Very useful on different tasks:
- Speech recognition;
- Spelling correction;

Applications

- Very useful on different tasks:
- Speech recognition;
- Spelling correction;
- Machine translation;

Applications

- Very useful on different tasks:
- Speech recognition;
- Spelling correction;
- Machine translation;
- LMs are useful in almost any tasks that deals with generating language.

Language Models

- N-gram based LMs;

Language Models

- N-gram based LMs;
- Log-linear LMs;

Language Models

- N-gram based LMs;
- Log-linear LMs;
- Neural LMs.

N-gram LM

- x is a sequence of words

N-gram LM

- x is a sequence of words
- $x = x_1, x_2, x_3, x_4, x_5$
  $=$ you, know, nothing, jon, snow

N-gram LM

- To compute the probability of a sentence

$$p(x) = p\left(x_1, x_2, \ldots, x_n\right) \tag{1}$$

N-gram LM

- To compute the probability of a sentence

$$p(x) = p\left(x_1, x_2, \ldots, x_n\right) \tag{1}$$

- We apply the chain rule:

$$p(x) = \prod_i p\left(x_i | x_1, \ldots, x_{i-1}\right) \tag{2}$$

N-gram LM

- To compute the probability of a sentence

$$p(x) = p(x_1, x_2, \ldots, x_n) \tag{1}$$

- We apply the chain rule:

$$p(x) = \prod_i p(x_i | x_1, \ldots, x_{i-1}) \tag{2}$$

- We limit the history with a Markov order:
$p(x_i | x_1, \ldots, x_{i-1}) \simeq p(x_i | x_{i-4}, x_{i-3}, x_{i-2}, x_{i-1})$

N-gram LM

- Chain rule:

$$p(x) = \prod_i p\left(x_i | x_1, \ldots, x_{i-1}\right) \quad (3)$$

N-gram LM

- We make a Markov assumption of conditional independence:

$$p\left(x_i \middle| x_1, \ldots, x_{i-1}\right) \simeq p\left(x_i \middle| x_{i-1}\right) \tag{4}$$

N-gram LM

- We make the Markov assumption:
  $p\left(x_i \middle| x_1, \ldots, x_{i-1}\right) \simeq p\left(x_i \middle| x_{i-1}\right)$

N-gram LM

- We make the Markov assumption: $p\left(x_i|x_1,\ldots,x_{i-1}\right) \simeq p\left(x_i|x_{i-1}\right)$
- If we do not observe the bi-gram $p(x_i|x_{i1})$ will be 0. This makes the probability of the sentence 0.

N-gram LM

- We make the Markov assumption: $p(x_i|x_1, \ldots, x_{i-1}) \simeq p(x_i|x_{i-1})$
- If we do not observe the bi-gram $p(x_i|x_{i1})$ will be 0. This makes the probability of the sentence 0.
- MLE

$$p_{\mathrm{MLE}}(x_i|x_{i-1}) = \frac{\text{count}(x_{i-1}, x_i)}{\text{count}(x_{i-1})} \tag{5}$$

N-gram LM

- We make the Markov assumption:
  $p\left(x_i|x_1,\ldots,x_{i-1}\right) \simeq p\left(x_i|x_{i-1}\right)$
- If we do not observe the bi-gram $p(x_i|x_{i1})$ will be 0. This makes the probability of the sentence 0.
- MLE

$$p_{\mathrm{MLE}}\left(x_i|x_{i-1}\right) = \frac{\text{count}\left(x_{i-1},x_i\right)}{\text{count}\left(x_{i-1}\right)} \tag{5}$$

- Laplace smoothing:

$$p_{\mathrm{add1}}\left(x_i|x_{i-1}\right) = \frac{\text{count}\left(x_{i-1},x_i\right)+1}{\text{count}\left(x_{i-1}\right)+V} \tag{6}$$

Log-linear LM

- $$p(y|x) = \frac{\exp \boldsymbol{w} \cdot \phi(x, y)}{\sum_{y' \in V_y} \exp w \cdot \phi\left(x, y'\right)} \tag{7}$$

Log-linear LM

- $$p(y|x) = \frac{\exp \boldsymbol{w} \cdot \phi(x,y)}{\sum_{y' \in V_y} \exp w \cdot \phi\left(x, y'\right)} \tag{7}$$
- y is the next word and V_y is the vocabulary;

Log-linear LM

- $$p(y|x) = \frac{\exp \boldsymbol{w} \cdot \phi(x, y)}{\sum_{y' \in V_y} \exp w \cdot \phi(x, y')} \tag{7}$$
- y is the next word and V_y is the vocabulary;
- x is the history;

Log-linear LM

- $$p(y|x) = \frac{\exp \boldsymbol{w} \cdot \phi(x, y)}{\sum_{y' \in V_y} \exp w \cdot \phi(x, y')} \tag{7}$$
- y is the next word and V_y is the vocabulary;
- x is the history;
- ϕ is a feature function that returns an n-dimensional vector;

Log-linear LM

- $$p(y|x) = \frac{\exp \boldsymbol{w} \cdot \phi(x, y)}{\sum_{y' \in V_y} \exp w \cdot \phi\,(x, y')} \tag{7}$$
- y is the next word and V_y is the vocabulary;
- x is the history;
- ϕ is a feature function that returns an n-dimensional vector;
- w are the model parameters.

Neural LM

- n-gram language models have proven to be effective in various tasks

Neural LM

- n-gram language models have proven to be effective in various tasks
- log-linear models allow us to share weights through features

Neural LM

- n-gram language models have proven to be effective in various tasks
- log-linear models allow us to share weights through features
- maybe our history is still too limited, e.g. n-1 words

Neural LM

- n-gram language models have proven to be effective in various tasks
- log-linear models allow us to share weights through features
- maybe our history is still too limited, e.g. n-1 words
- we need to find useful features

Feed-forward NLM

- With NN we can exploit distributed representations to allow for statistical weight sharing.

Feed-forward NLM

- With NN we can exploit distributed representations to allow for statistical weight sharing.
- How does it work:

Feed-forward NLM

- With NN we can exploit distributed representations to allow for statistical weight sharing.
- How does it work:
  1. each word is mapped to an embedding: an m-dimensional feature vector;

Feed-forward NLM

- With NN we can exploit distributed representations to allow for statistical weight sharing.
- How does it work:
  1. each word is mapped to an embedding: an m-dimensional feature vector;
  2. a probability function over word sequences is expressed in terms of these vectors;

Feed-forward NLM

- With NN we can exploit distributed representations to allow for statistical weight sharing.
- How does it work:
  1. each word is mapped to an embedding: an m-dimensional feature vector;
  2. a probability function over word sequences is expressed in terms of these vectors;
  3. we jointly learn the feature vectors and the parameters of the probability function.

Feed-forward NLM

- Similar words are expected to have similar feature vectors: (dog,cat), (running,walking), (bedroom,room)

Feed-forward NLM

- Similar words are expected to have similar feature vectors: (dog,cat), (running,walking), (bedroom,room)
- With this, probability mass is naturally transferred from (1) to (2):

Feed-forward NLM

- Similar words are expected to have similar feature vectors: (dog,cat), (running,walking), (bedroom,room)
- With this, probability mass is naturally transferred from (1) to (2):
- The cat is walking in the bedroom.

Feed-forward NLM

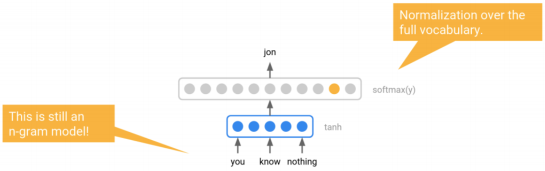
- Similar words are expected to have similar feature vectors: (dog,cat), (running,walking), (bedroom,room)
- With this, probability mass is naturally transferred from (1) to (2):
- The cat is walking in the bedroom.
- The dog is running in the room.

Feed-forward NLM

- Similar words are expected to have similar feature vectors: (dog,cat), (running,walking), (bedroom,room)
- With this, probability mass is naturally transferred from (1) to (2):
- The cat is walking in the bedroom.
- The dog is running in the room.
- Take-away message:
  The presence of only one sentence in the training data will increase the probability of a combinatorial number of neighbours in sentence space.

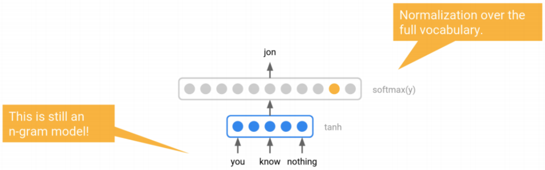
Feed-forward NLM

- FF-LM



Feed-forward NLM

- FF-LM

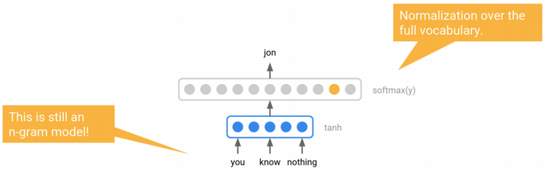


- $\boldsymbol{E}_{\text{you}}, \boldsymbol{E}_{\text{know}}, \boldsymbol{E}_{\text{nothing}} \in \mathbb{R}^{100}$

$$\begin{aligned} \boldsymbol{x} &= [\boldsymbol{E}_{\text{you}}; \boldsymbol{E}_{\text{know}}; \boldsymbol{E}_{\text{nothing}}] \in \mathbb{R}^{300} \\ \boldsymbol{y} &= \boldsymbol{W}_3 \tanh(\boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{W}_2 \boldsymbol{x} + \boldsymbol{b}_2 \end{aligned} \tag{8}$$

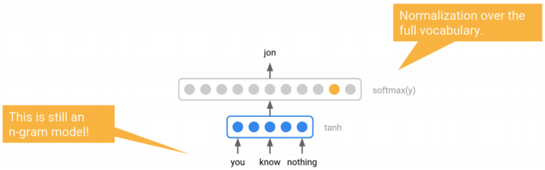
Feed-forward NLM

- FF-LM



Feed-forward NLM

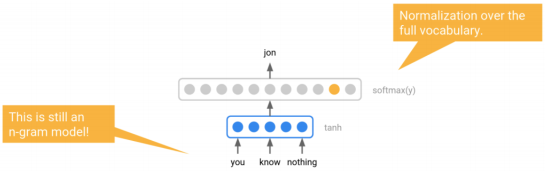
- FF-LM



- The non-linear activation functions perform feature combinations that a linear model cannot do;

Feed-forward NLM

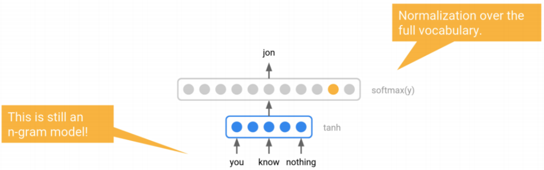
- FF-LM



- The non-linear activation functions perform feature combinations that a linear model cannot do;
- End-to-end training on next word prediction.

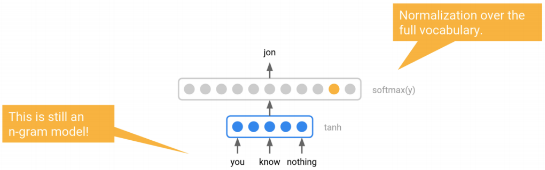
Feed-forward NLM

- FF-LM



Feed-forward NLM

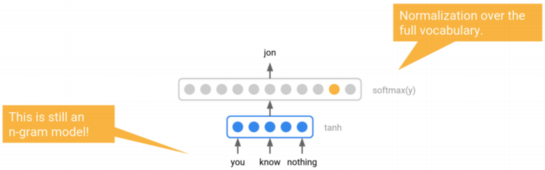
- FF-LM



- We now have much better generalisation, but still a limited history/context.

Feed-forward NLM

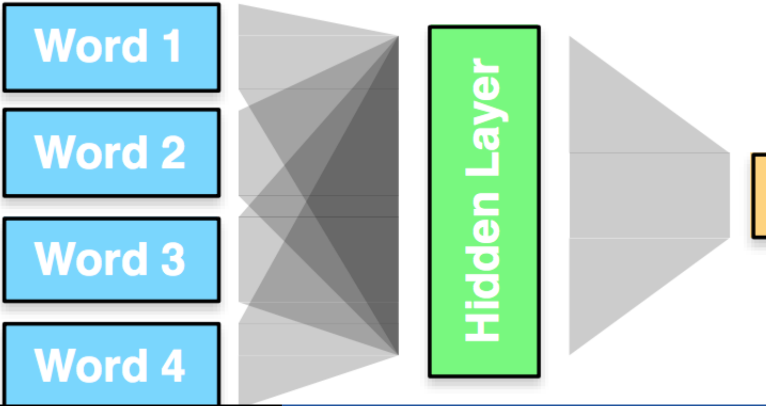
- FF-LM



- We now have much better generalisation, but still a limited history/context.
- Recurrent neural networks have unlimited history

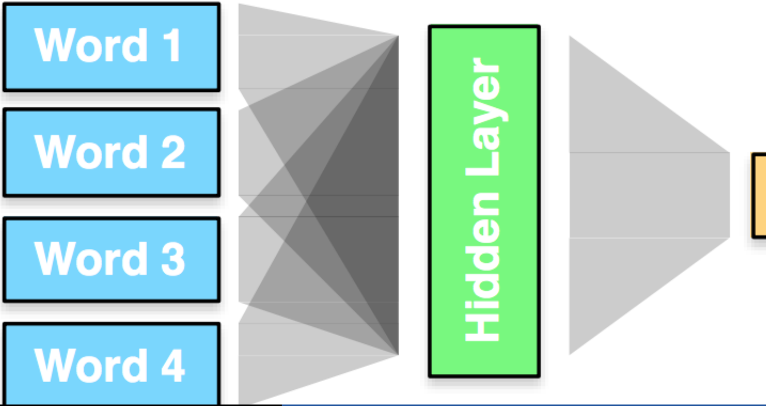
Feed-forward NLM

- FF-LM



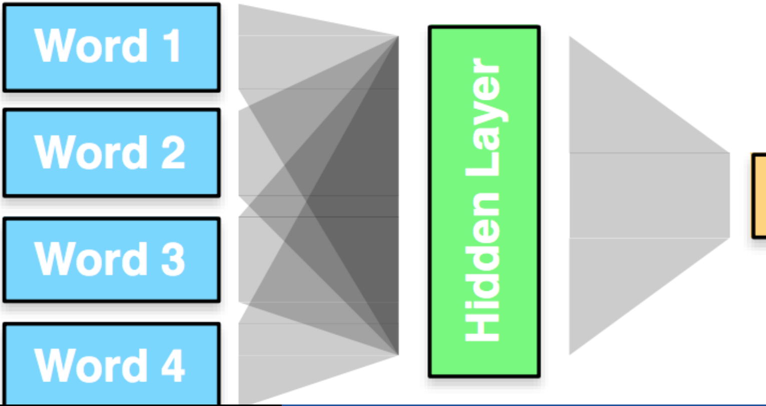
Feed-forward NLM

- FF-LM



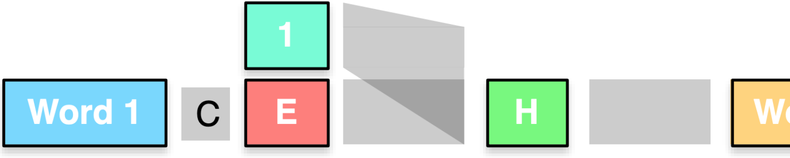
Feed-forward NLM

- FF-LM



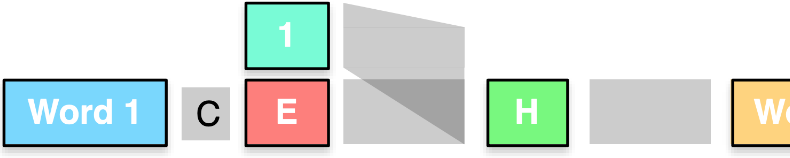
RNN NLM

- RNN-LM



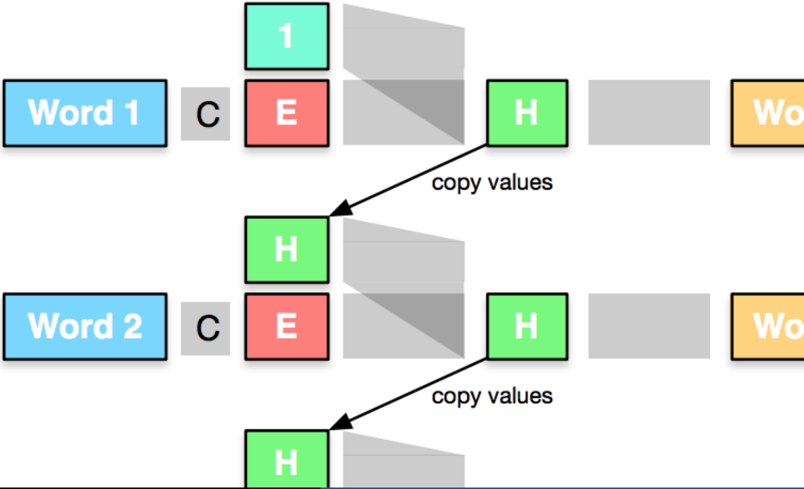
RNN NLM

- RNN-LM



- Start: predict second word from first

RNN NLM



Outline

1. Language Modelling
2. Variational Auto-encoder for Sentences
   - Model

-

References I