

Nils Reiter, nils.reiter@uni-koeln.de

June 8, 2021 (Summer term 2021)

Recap

Bag of words

- Count words, disregard their order
- ▶ Document \rightarrow vector \rightarrow input for neural network
- scikit-learn: CountVectorizer

Recap

Bag of words

- Count words, disregard their order
- Document \rightarrow vector \rightarrow input for neural network
- scikit-learn: CountVectorizer

Overfitting

- Good performance on training data
- Less performance on real-world data
- No strict, deterministic decision
- Regularization: Add something to loss function
- Dropout: Randomly remove edges during training, force the network to create redundancies

Recap

Bag of words

- Count words, disregard their order
- \blacktriangleright Document \rightarrow vector \rightarrow input for neural network
- scikit-learn: CountVectorizer

Overfitting

- Good performance on training data
- Less performance on real-world data
- No strict, deterministic decision
- Regularization: Add something to loss function
- Dropout: Randomly remove edges during training, force the network to create redundancies

Exercise 7

Reiter

Administration

- Modulprüfungen
 - Modul: Angewandte Linguistische Datenverarbeitung
 - Anmeldung: Bis 30.07.
 - Abmeldung: Bis 14.09.
 - Modus: Hausarbeit mit praktischem Anteil/Experiment
 - ► Kann, muss aber nicht, Bezug zum HS Computational Literary Studies haben
 - ...oder zu einem Modul-HS aus den letzten Semestern
 - Abgabedatum: 15.09.
 - Abweichung in Absprache möglich



Input Representation

Embeddings

Implementing Embeddings in Keras

Exercise

Section 1

Input Representation

Last Week: Bag of Words

- Disregard word order
 - Makes things a lot easier
 - But it's not how language works

Last Week: Bag of Words

- Disregard word order
 - Makes things a lot easier
 - But it's not how language works
- Example
 - »This remake was even greater than the original.«
 - »Deciding to make the script into a 3D movie led to an even greater failure.«

Last Week: Bag of Words

- Disregard word order
 - Makes things a lot easier
 - But it's not how language works
- Example
 - »This remake was even greater than the original.«
 - »Deciding to make the script into a 3D movie led to an even greater failure.«
- ▶ How could we represent words in order and context with feed-forward neural networks?



- Enumerate all words and replace the strings with index numbers
- Use n words at once as input features

- Enumerate all words and replace the strings with index numbers
- Use n words at once as input features

Example

▶ »The best soundtrack ever to anything: [...]«

- Enumerate all words and replace the strings with index numbers
- Use n words at once as input features

Example

»The best soundtrack ever to anything: [...]«
 Bag of Words

$$text = \left[\begin{array}{rrr} a & 5 \\ an & 2 \\ animal & 0 \end{array} \right]$$

- Enumerate all words and replace the strings with index numbers
- Use n words at once as input features

Example

»The best soundtrack ever to anything: [...]«
 Bag of Words
 Sequential word indices

$$text = \begin{bmatrix} a & 5\\ an & 2\\ animal & 0 \end{bmatrix} \qquad text = \begin{bmatrix} the & 5\\ best & 124\\ soundtrack & 15 \end{bmatrix}$$

In a feed-forward neural network



In a feed-forward neural network



In a feed-forward neural network



Problems

- Fixed word positions No connection between these sentences:
 - »This remake was even greater than the original.« vs.
 - »Arguably, this remake was even greater than the original.«

7

-> Convolution CNN

Problems

- Fixed word positions No connection between these sentences:
 - »This remake was even greater than the original.« vs.
 - »Arguably, this remake was even greater than the original.«
- No generalization across small variations 2 -> Embedde w
 - »[...] led to an even bigger failure«

Problems

- Fixed word positions No connection between these sentences:
 - »This remake was even greater than the original.« vs.
 - »Arguably, this remake was even greater than the original.«
- No generalization across small variations
 - ▶ »[...] led to an even bigger failure«
- No connection irrespective of the position

Section 2

Embeddings

Motivation

An embedding is a mapping of words or documents to vectors

Things are)embedded(in a vector space

Motivation

- An embedding is a mapping of words or documents to vectors
 - Things are)embedded(in a vector space
- Why do we need this?
 - Classically, words are discrete symbols
 - ▶ For neural networks, each word is replaced by a word index

Motivation

- An embedding is a mapping of words or documents to vectors
 - Things are >embedded(in a vector space
- Why do we need this?
 - Classically, words are discrete symbols
 - For neural networks, each word is replaced by a word index
- We can do better
 - Representing a word as a vector allows calculating similarity between words
 - ▶ If the embedding works well, similarity between word *meanings*

Documents as vectors

Document	Dog	Cat
1	10	2
2	5	0
3	1	5
4	5	5

Documents as vectors

Document	Dog	Cat
1	10	2
2	5	0
3	1	5
4	5	5



Documents as vectors

Document	Dog	Cat
1	10	2
2	5	0
3	1	5
4	5	5

- Document-term-matrix Sparsely populated!
- Count vector
- Similarity based on vector angle

1	•		
			\



Co-occurrence windows: Count how often a word co-occurs with every other word

- \blacktriangleright >Vocabulary V: Set of words in entire corpus
- Results in $|V| \times |V|$ matrix
- Mostly zeros (because of the Zipf distribution)

Words as vectors

► Co-occurrence windows: Count how often a word co-occurs with every other word

- > Vocabulary V: Set of words in entire corpus
- Results in $|V| \times |V|$ matrix
- Mostly zeros (because of the Zipf distribution)
- Singular value decomposition (SVD): Mathematically reduce dimensions

Word Embeddings (after 2013)

> fasted

Bojanowski et al. (2016), Mikolov et al. (2013), and Pennington et al. (2014), ...

- Dense representations of words in vector space
- ▶ Word vectors: Weights learned by a simple neural network with a classification target

wordluce

▶ word2vec: Given word w_i , how likely is it that w_j appears in its context?

Idea

- Embeddings are learned using a neural network
- Classification task: Given a word, predict its context words
 - Training data in abundance
- Use learned weights as embeddings

Slove

- Existing (pre-trained) embeddings can be plugged in
- Specific embeddings can be trained, just like all other weights

- Existing (pre-trained) embeddings can be plugged in
- Specific embeddings can be trained, just like all other weights

Pre-trained embeddings

- Glove (Stanford University): https://nlp.stanford.edu/projects/glove/
- FastText (facebook research): https://fasttext.cc (multiple languages)

- Existing (pre-trained) embeddings can be plugged in
- Specific embeddings can be trained, just like all other weights

Pre-trained embeddings

- Glove (Stanford University): https://nlp.stanford.edu/projects/glove/
- FastText (facebook research): https://fasttext.cc (multiple languages)

adventure (0.0292 -0.0269 0.0273 0.0792 -0.0617 0.1370 -0.0628 0.0420 0.0743 0.0979 -0.0136 0.0488 -0.0267 -0.0227 0.0592 0.0410 0.0314 0.0378 -0.0455 0.0616 -0.0380 0.0232 -0.0218 0.0000 -0.0699 -0.1327 -0.0393 0.0467 0.0413 0.0089 -0.0046 0.0372 -0.0590 0.0740 0.0214 0.0625 0.0067 -0.0063 0.0218 -0.0447 -0.0298 0.0186 -0.0207 0.0158 -0.0508 -0.0297 -0.0807 -0.0619 -0.0194 -0.0153 0.0909 -0.0037 0.0999 -0.0110 ...



This changes the input data shape

This	is	а	great	adventure	
------	----	---	-------	-----------	--

This changes the input data shape



This changes the input data shape



This is a matrix!

I.e., by embedding tokens into a vector space, we have changed the shape of our data from 1D to 2D

Fixed Length of Input

- Our input now consists of a matrix (per instance)
- Matrix size needs to be predefined
 - Embedding dimension: Parameter we can set freely
 - Length: To be set on training data

Fixed Length of Input

- Our input now consists of a matrix (per instance)
- Matrix size needs to be predefined
 - Embedding dimension: Parameter we can set freely
 - Length: To be set on training data
- Input length
 - ▶ This parameter controls how long sentences (or texts) can be
 - It's a hard limit

Fixed Length of Input

- Our input now consists of a matrix (per instance)
- Matrix size needs to be predefined
 - Embedding dimension: Parameter we can set freely
 - Length: To be set on training data
- Input length
 - This parameter controls how long sentences (or texts) can be
 - It's a hard limit
- Padding
 - Extend shorter inputs so that they have the same length
 - Truncate longer inputs
 - Keras: Function tensorflow.keras.preprocessing.sequence/pad_sequences(...)

Section 3

Implementing Embeddings in Keras

Implementing Embeddings in Keras

Two relevant new layers

- tensorflow.python.keras.layers.Embedding()
- tensorflow.python.keras.layers.Flatten()

Preparations

- tensorflow.keras.preprocessing.text.Tokenizer()
- tensorflow.keras.preprocessing.text.text_to_word_sequence()
- tensorflow.keras.preprocessing.sequence.pad_sequences()

tensorflow.python.keras.layers.Embedding(...)

- Must be the first layer of the model
- »Turns positive integers (indexes) into dense vectors of fixed size«

, Divensione

tensorflow.python.keras.layers.Embedding(...)

- Must be the first layer of the model
- > »Turns positive integers (indexes) into dense vectors of fixed size«
- Parameters
 - input_dim: Size of the vocabulary (i.e., how many words to distinguish)
 - output_dim: How many elements to word vectors have?
 - input_length: Length of input vectors (e.g., sentences)

tensorflow.python.keras.layers.Embedding(...)

- Must be the first layer of the model
- »Turns positive integers (indexes) into dense vectors of fixed size«

Parameters

- input_dim: Size of the vocabulary (i.e., how many words to distinguish)
- output_dim: How many elements to word vectors have?
- input_length: Length of input vectors (e.g., sentences)

Pre-trained embeddings

Documentation: https://keras.io/examples/nlp/pretrained_word_embeddings/

Parameters

embeddings_initializer=keras.initializers.Constant(embedding_matrix)

trainable=False

embedding_matrix is a numpy matrix that contains the vectors loaded from a file

Flatten

- Network structure so far: Only 1-dimensional vectors
- Result of embedding layer: matrices (2D)
- Flatten layer: Combine all rows of a matrix to a long vector
- layers.Flatten()
 - Usually used after an embedding layer





Preparations

- 1. Tokenizer
 - Establish vocabulary
 - Assign each type an integer number
- 2. Map token sequences to integer sequences
- 3. Padding
 - Ensure that all sequences have the same length by truncating or adding

Full Example

```
tokenizer = Tokenizer()
2 tokenizer.fit on texts(train texts)
3 vocab size = len(tokenizer.word index) + 1
4 train_texts = tokenizer.texts_to_sequences(train_texts)
5
  MAX LENGTH = max(len(train ex) for train ex in train texts)
6
7
  train texts = pad sequences(train texts, maxlen=MAX LENGTH, padding="post")
8
9
10 model = models.Sequential()
  model.add(lavers.Input(shape=(MAX LENGTH)))
11
12 model.add(layers.Embedding(vocab_size, 100, input_length=MAX_LENGTH))
13 model.add(layers.Flatten())
14 model.add(lavers.Dense(10, activation="sigmoid"))
  model.add(layers.Dropout(0.5))
15
  model.add(layers.Dense(1, activation="sigmoid"))
16
17
18 model.summary()
```

Section 4

Exercise

Exercise

Exercise 08

https://github.com/idh-cologne-deep-learning/exercise-08