Recep Oğuz Araz
ELEC350 Independent Study
Prof. Engin Erzin
28.10.2020

Report I

This report includes notes from the first 2 lectures of CS224 course on word embeddings and some discussion on our implementation of Baidu's DeepSpeech 2.

Word Embeddings

Represents the meanings of words with vectors.

Meaning: Idea that is represented by a word, phrase, …

WordNet is an early attempt to capture such relations. Thesaurus, containing lists of synonyms and hypernyms (relationships between words). Distinctions between words in different sentences are captured but synonym lists miss nuances. It is built with human labor.

It can not keep up with trends (New terms such as Wicked, badass).

(NLTK: basic tools for simple tasks. Tokenizer, capitalizer…).

Traditional NLP ➔ words are represented by discrete symbols. One-hot vectors.

Suffixes increase dimensions. Paternal, paternalism, paternalistically. (TR: Ev, Evler, Evden, Evimiz, Evcilik…)

One-hot vectors are orthogonal, cannot define similarities.

Distributional Semantics: Meaning of words are given by words that occur frequently close-by.

Small but dense vectors (min.50 dim to 4000 dim).

Need lots of text to construct.

Word2Vec: Initialize by randomizing each word vector, maximize the objective function. Probabilities are estimated with softmax calculations on contexts.

Notes for the Model:

Use GENSIM for vector visualizations.

Use row vectors for words. (Simple calculations.)

For optimization use Stochastic Gradient Descent (Mini-batch processing specifically) rather than Gradient Descent.

1. Less noisy than Batch Processing.
2. Exploit GPU power.

Recep Oğuz Araz
ELEC350 Independent Study
Prof. Engin Erzin
28.10.2020

Window size should be a power of two (best match with GPU hardware).

Sparse parameter update (??)

Use negative sampling in training: Additional efficiency. (Binary logistic regression)

For each epoch shuffle the data.

Word Embeddings will result in accuracy increase.

GLoVe (Skip-grams, CBOW)

Will we use word embeddings?

Where will we use word embeddings in? Language model after the recognizer?

Do we have a corpus?

What we have:

LDC2006S33: 500 minutes of labeled sentences. (40 sentence per speaker).

Mozilla Common Voice Turkish Dataset: 20 hours of labeled data. Includes Noisy recordings and some have poor Turkish.

Our Model:

Baidu's DeepSpeech 2 has 8-11 layers including many bi-directional layers as they have 12000 hours of data. We will limit our layers to something smaller.

Each utterance $x^{(i)}$ is a time-series of length $T^{(i)}$ where every time-slice is a vector of audio features. The original paper uses power normalized audio clips as features. However, we are planning of using Mel-Spectrogram coefficients.

Goal of the RNN is to convert an input sequence $x^{(i)}$ to a transcription $y^{(i)}$.

Our model will label the input sequence with graphemes without segmentation and yield a final transcription for an audio signal without any segmentation or aligning using Connectionist Temporal Classification Loss (CTC).