Recognizing Textual Entailment Capstone Project

Build a Natural Language Inference

Kirils Mensikovs
April 19, 17

Proposal

I want to concentrate on the problem of sentence encoding - extracting a real-valued vector to represent the meaning of a sentence. To get understand words meaning I will learn Textual Entailments (TE). In Natural Language Processing (NLP) TE is a directional relation between text fragments. The relation holds whenever the truth of one text fragment follows from another text. TE is a generalization of many tasks in NLP. If you have a system that is good at recognizing TE, it should be easier to build good systems for Information Retrieval (IR), Question Answering (QA), Paraphrase Recognition (PR), Information Extraction (IE) and summarization. Typically entailment is used as part of a large system.

Domain Background

Search engines, chat bots and assistants such as Alexa or Google Home have been using Natural Language Processing (NLP) to better understand the meaning of words and context of the situation to make the intelligent answer to the people. Researchers have been used different techniques to make the computer understand people. Starting with the simple bag of words, tf-idf models and applying manually labeled frames. But all those algorithms don't provide sufficient results.

A characteristic of natural language is that there are many different ways to state what you want to say. Several meanings can be contained in a single text and that the same meaning can be expressed by different text.

Problem Statement

Given two text fragments, one named text (t) and the other named hypothesis (h), respectively. The task consists in recognizing whether the hypothesis can be inferred from the text. TE has a three-class balanced classification problem over sentence pairs: *entailment*, *neutral* and *contradiction*.

Example:
Text: *Your gift is appreciated by each and every student who will benefit from your generosity.*
Hypothesis: *Hundreds of students will benefit from your generosity.*
Label: *neutral*

Datasets and Inputs

I will use the Multi-Genre NLI Interface (MultiNLI) corpus to create the model. MultiNLP is a crowd-sourced collection of 433k sentences pairs annotated with textual entailment information. The corpus contains from genres of spoken and written text.

Solution statement

I will build the bilateral multi-perspective matching (BiMPM) model [Wang, Hamza and Florian, 2017]. The model will matches sentence and hypothesis in two directions. From the sentence follow the hypothesis and from the hypothesis follow the sentence. In each direction, the model matches the two sentences from multiple perspectives. The model contains the following five layers.

Word representation layer – construct the d-dimensional vector from the word embedding GloVe[Pennington et al., 2014] vector for each individual word.

Context representation layer – incorporate contextual information into the representation of each time step.

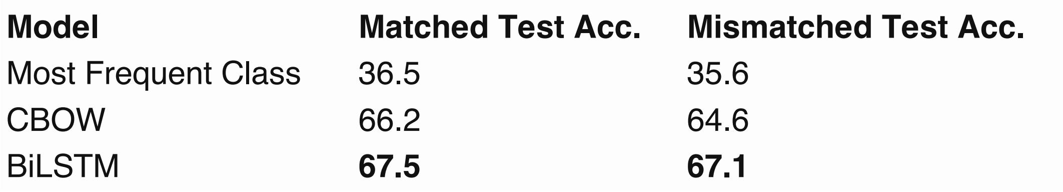
Matching layer – compare each contextual embedding of one sentence against all contextual embeddings of the other sentence.

Aggregation layer – aggregate two sequences of matching vectors into a fix-length matching vector.

Prediction layer – evaluate the probability distribution Pr(y|P,Q), Pr(y|P,Q), Pr(y|P,Q).

Benchmark model

A study found humans to be in agreement on the dataset 95.25% of the time [Bos and Markert, January 2005]. I will use the RepEval 2017 Shared task [https://repeval2017.github.io/shared/]. They trained three models and evaluate them. I would like to get at least 67% accuracy in my tests.



Model	Matched Test Acc.	Mismatched Test Acc.
Most Frequent Class	36.5	35.6
CBOW	66.2	64.6
BiLSTM	**67.5**	**67.1**

Evaluation metrics

I will measure the performance of the application on matched and mismatched dataset that is provided by RepEval 2017 Shared task.

Project design

Keras and Tensorflow will be used to train and test the model. The code will be developed in IPython notebook.

Requirements:

- Python 2.7
- Nympy
- Tensorflow
- Keras
- IPython notebook