Extracting Vector Representation of DBPedia Properties from word2vec

Abstract

Vector representation of words has been learned by word2vec and useful in many natural language processing and information retrieval. However, the advantages that can be utilized by applying word2vec have not been applied for Dbpedia data. In this paper, word2vec for Dbpedia properties is proposed. The main goals of Extracting Vector Representation of Dbpedia Properties are to disambiguate terms given by users with the intention of finding their corresponding resources from the Dbpedia and to apply Word2Vec technique to find missing information.

Introduction

DBpedia (from "DB" for "database") is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. DBpedia allows you to ask sophisticated queries against Wikipedia, and to link the different data sets on the Web to Wikipedia data.

Word2Vev Tool

Word2Vec tool provides an efficient implementation of the continuous bag-of-words (predicts a missing word given a window of context words or word sequence) and skip-gram (using a word to predict a target context) architectures for computing vector representations of words. These representations can be subsequently used in many natural language processing, machine learning applications and for further research. The **word2vec** tool takes a text corpus as input and produces the word vectors as output. It first constructs a vocabulary from the training text data and then learns vector representation of words.

A simple way to investigate the learned representations is to find the closest words for a user-specified word. The **distance** tool serves that purpose. For example, if you enter 'german', **distance** will display the most similar words and their distances to 'german'.

Word2vec is a two-layer neural net that processes text. Its input is a text corpus and its output is a set of vectors: feature vectors for words in that corpus.

It was recently shown that the word vectors capture many linguistic regularities, for example vector operations **vector('Paris') - vector('France') + vector('Italy')** results in a vector that is very close to **vector('Rome')**, and **vector('king') - vector('man') + vector('woman')** is close to **vector('queen')**

Motivation

The main aim of this project is to introduce techniques that can be used for learning high-quality word vectors from Dbpedia.

Related Work

Representation of words as continuous vectors has a long history. A very popular model architecture for estimating neural network language model (NNLM), where a feedforward neural network with a linear projection layer and a non-linear hidden layer was used to learn jointly the word vector representation and a statistical language model.

Another interesting architecture of NNLM was presented in [3, 4], where the word vectors are first learned using neural network with a single hidden layer. The word vectors are then used to train the NNLM. Thus, the word vectors are learned even without constructing the full NNLM.

It was later shown that the word vectors can be used to significantly improve and simplify many NLP applications. Estimation of the word vectors itself was performed using different model architectures and trained on various corpora, and some of the resulting word vectors were made available for future research and comparison.

Many different types of models were proposed for estimating continuous representations of words, including the well-known Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA). Tomas Mikolov and his team focused on distributed representations of words learned by neural networks, as it was previously shown that they perform significantly better than LSA for preserving linear regularities among words.

Feedforward Neural Net Language Model (NNLM)

The probabilistic feedforward neural network language model consists of input, projection, hidden and output layers. At the input layer, N previous words are encoded using 1-of-V coding, where V is size of the vocabulary. The input layer is then projected to a projection layer P that has dimensionality N _ D, using a shared projection matrix. As only N inputs are active at any given time, composition of the projection layer is a relatively cheap operation.

Recurrent Neural Net Language Model (RNNLM)

Recurrent neural network based language model has been proposed to overcome certain limitations of the feedforward NNLM, such as the need to specify the context length (the order of the model N), and because theoretically RNNs can efficiently represent more complex patterns than the shallow neural networks. The RNN model does not have a projection layer; only input, hidden and output layer. What is special for this type of model is the recurrent matrix that connects hidden layer to itself, using time-delayed connections. This allows the recurrent model to form some kind of short term memory, as information from the past can be represented by the hidden layer state that gets updated based on the current input and the state of the hidden layer in the previous time step.

Implementation

Continuing