### **Week 7 : NLP: Word Embeddings, Semantic Analysis, text 5: SLP selected chapters**

1. [Blog and Code](https://towardsdatascience.com/visualizing-topic-models-with-scatterpies-and-t-sne-f21f228f7b02) on Various Word embedding Methods
2. [Word embeddings](https://machinelearningmastery.com/develop-word-embeddings-python-gensim/) with Gensim .
3. LSA, PLSA, LDA. [Latent Dirichlet Allocation](http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf), David Blei, et al. [Topic modeling]
4. [Minimizing Fake News](https://blog.machinebox.io/detect-fake-news-by-building-your-own-classifier-31e516418b1d)
5. [A Quick Guide to Fake News Detection on Social Media](https://www.kdnuggets.com/2017/10/guide-fake-news-detection-social-media.html),
   1. [academic version](https://arxiv.org/abs/1708.01967) of **Fake News Detection on Social Media: A Data Mining Perspective**
   2. another related article by Kai Shu: [**Studying Fake News via Network Analysis: Detection and Mitigation**](https://arxiv.org/abs/1804.10233)
6. [**Getting Real About Fake News**](https://www.quora.com/What-are-some-datasets-about-fake-news) **:** [**code**](https://www.kaggle.com/mrisdal/fake-news)
7. [**Detecting Fake News**](https://www.kdnuggets.com/2017/04/machine-learning-fake-news-accuracy.html)
8. [**https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/**](https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/)
9. <https://medium.com/analytics-vidhya/demystifying-bert-the-groundbreaking-nlp-framework-8e3142b3d366>

**Reading notes:**

1. Various Word embedding Methods

* Binary Encoding.
* TF Encoding.
* TF-IDF Encoding.
* Latent Semantic Analysis Encoding.
* **Word2Vec Embedding**.

2. [Word embeddings](https://machinelearningmastery.com/develop-word-embeddings-python-gensim/) with Gensim:

* How to train word2vec word embedding model on text data.

from gensim.models import Word2Vec

# define training data

sentences = [['this', 'is', 'the', 'first', 'sentence', 'for', 'word2vec'],

['this', 'is', 'the', 'second', 'sentence'],

['yet', 'another', 'sentence'],

['one', 'more', 'sentence'],

['and', 'the', 'final', 'sentence']]

# train model

model = Word2Vec(sentences, min\_count=1)

* Visualize a trained word embedding model using PCA

# fit a 2d PCA model to the vectors

X = model[model.wv.vocab]

pca = PCA(n\_components=2)

result = pca.fit\_transform(X)

# create a scatter plot of the projection

pyplot.scatter(result[:, 0], result[:, 1])

words = list(model.wv.vocab)

for i, word in enumerate(words):

pyplot.annotate(word, xy=(result[i, 0], result[i, 1]))

pyplot.show()

* Load pre-trained word2vec and GloVe word embedding models from Google and Stanford

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | from gensim.models import KeyedVectors  # load the google word2vec model  filename = 'GoogleNews-vectors-negative300.bin'  model = KeyedVectors.load\_word2vec\_format(filename, binary=True)  # calculate: (king - man) + woman = ?  result = model.most\_similar(positive=['woman', 'king'], negative=['man'], topn=1)  print(result) |

from gensim.scripts.glove2word2vec import glove2word2vec

glove\_input\_file = 'glove.txt'

word2vec\_output\_file = 'word2vec.txt'

glove2word2vec(glove\_input\_file, word2vec\_output\_file)

3. [Minimizing Fake News](https://blog.machinebox.io/detect-fake-news-by-building-your-own-classifier-31e516418b1d)

The author thinks artificial intelligence presents us with an opportunity to tackle this fake news problem at scale.

The author uses classification box technology to train the model to detect bias in news.

To decide what’s neutral and what’s biased, the author suggests a fake news classifier Fakebox can be used.

One key principle is that each class should have more or less the same number of examples.

4. [A Quick Guide to Fake News Detection on Social Media](https://www.kdnuggets.com/2017/10/guide-fake-news-detection-social-media.html),

The author thinks fake news on social media has its unique characteristics. For example, malicious accounts can be easily and quickly created to boost the spread of fake news. Also, users on social media tend to form groups containing like-minded people where they are likely to polarize their opinions, resulting in an echo chamber effect.

The author categorized existing algorithms for fake news detection can be generally as (i) News Content Based and (ii) Social Context Based:

1. News content based approaches focus on extracting various features in fake news content, including knowledge-based and style-based.
2. Knowledge-based approaches aim to using external sources to fact-check the truthfulness of the claims in news content. Style-based approaches try to detect fake news by capturing the manipulators in the writing style

Social context based approaches aim to utilize user social engagements as auxiliary information to help detect fake news:

1. Stance-based approaches utilize users’ viewpoints from relevant post contents to infer the veracity of original news articles.
2. Propagation-based approaches reason about the relations of relevant social media posts to guide the learning of credibility scores by propagating credibility values between users, posts, and news.

5. [academic version](https://arxiv.org/abs/1708.01967) of Fake News Detection on Social Media:

In the article, the author studies social media fake news via network analysis. The author defined 3 dimensions in the news dissemination ecosystem: content dimension, social dimension, temporal dimension. Also, the author introduced network properties Echo Chambers Filter Bubbles ect.. And, network types:

Homogeneous networks including friendship network, diffusion network and credibility network.

Heterogeneous networks including knowledge network, stance network and interaction network.

Then, the author formulate these network phenomena with mathematics, and create models for fake news prediction and mitigation.

6. [Detecting Fake News](https://www.kdnuggets.com/2017/04/machine-learning-fake-news-accuracy.html)

The author attempted to build a model that can differentiate between “Real” news vs “Fake” news. The author used All Sides scraping full text of a total of 5279 articles.

* The model is a Naive Bayes classifier with text transformation count vectorizer vs tfidf vectorizer.
* The author used sklearn GridSearch functionality to tune the parameters and efficiently execute the task.
* Count vectorizer worked better than tfidf vectorizer in the author’s model.
* The logic the author used to determine if the work is ‘fake’ or ‘real’, if a word shows up a bunch in “fake” articles and rarely in “real” articles then its fake to real ratio score will be pretty high.
* The model’s cross-validated accuracy score is 91.7%, which is beyond the author’s expectations.

6. [Various Word embedding Methods](https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/)

### 1) Frequency based Embedding

There are generally three types of vectors that we encounter under this category.

* Count Vector
* TF-IDF Vector
* Co-Occurrence Vector (It preserves the semantic relationship between words)

2) Prediction based Embedding

Word2vector is a combination of CBOW(Continuous bag of words) and Skip-gram model

Comparison: The objective function in MLP is a MSE(mean square error) whereas in CBOW it is negative log likelihood of a word given a set of context i.e -log(p(wo/wi)).

8. [Demystifying BERT](https://www.analyticsvidhya.com/blog/2019/09/demystifying-bert-groundbreaking-nlp-framework/)

BERT stands for Bidirectional Encoder Representations from Transformers, the most influential one in the recent NLP framework according to the author.

* BERT is based on transformer architecture.
* BERT is pretrained on a large text corpus.
* Bidirectional means BERT learns from both left and right side of the token’s context.
* Word2vec limitations
  + Using shallow language model
  + Didn’t take context into account
* Transfer Learning in NLP = Pre-Training and Fine-Tuning
* OpenAI’s GPT validated the robustness and usefulness of the Transformer architecture by achieving multiple State-of-the-Arts.
* BERT Base architecture has the same model size as OpenAI’s GPT for comparison purposes: All of these Transformer layers are **Encoder**-only blocks.
* The developers of BERT have added a set of rules, every its input representation is constructed by summing the corresponding **token**, **segment**, and **position** embeddings.
* A deep bidirectional model is strictly more powerful than either a left-to-right model or the shallow concatenation of a left-to-right and a right-to-left model, that is where BERT greatly improves upon both GPT and ELMo.
* [Bert As Service](https://github.com/hanxiao/bert-as-service) -- allow us use BERT to extract encodings for each sentence in just two lines of code.
* BERT can be used for text classification and sentence prediction.