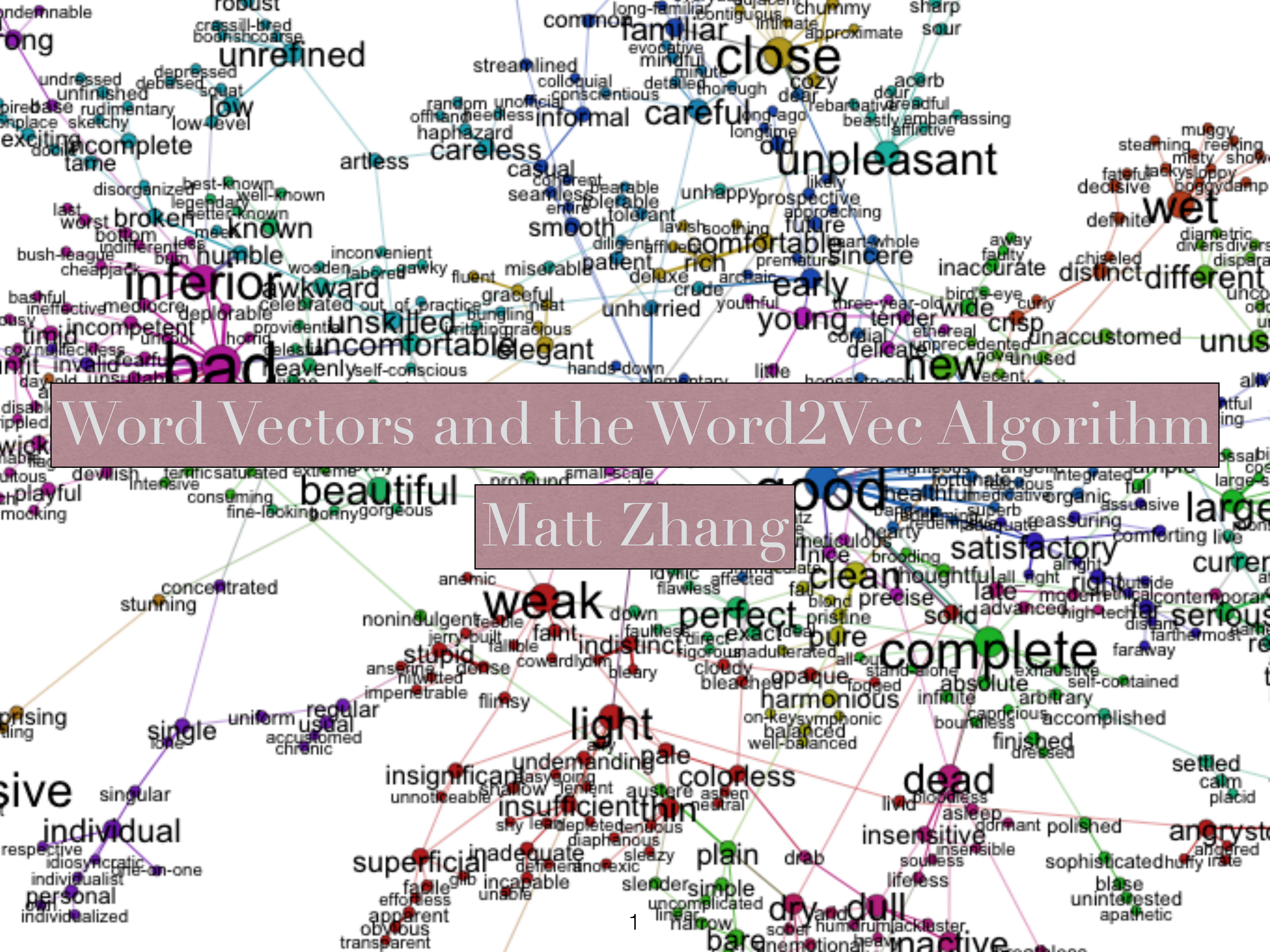
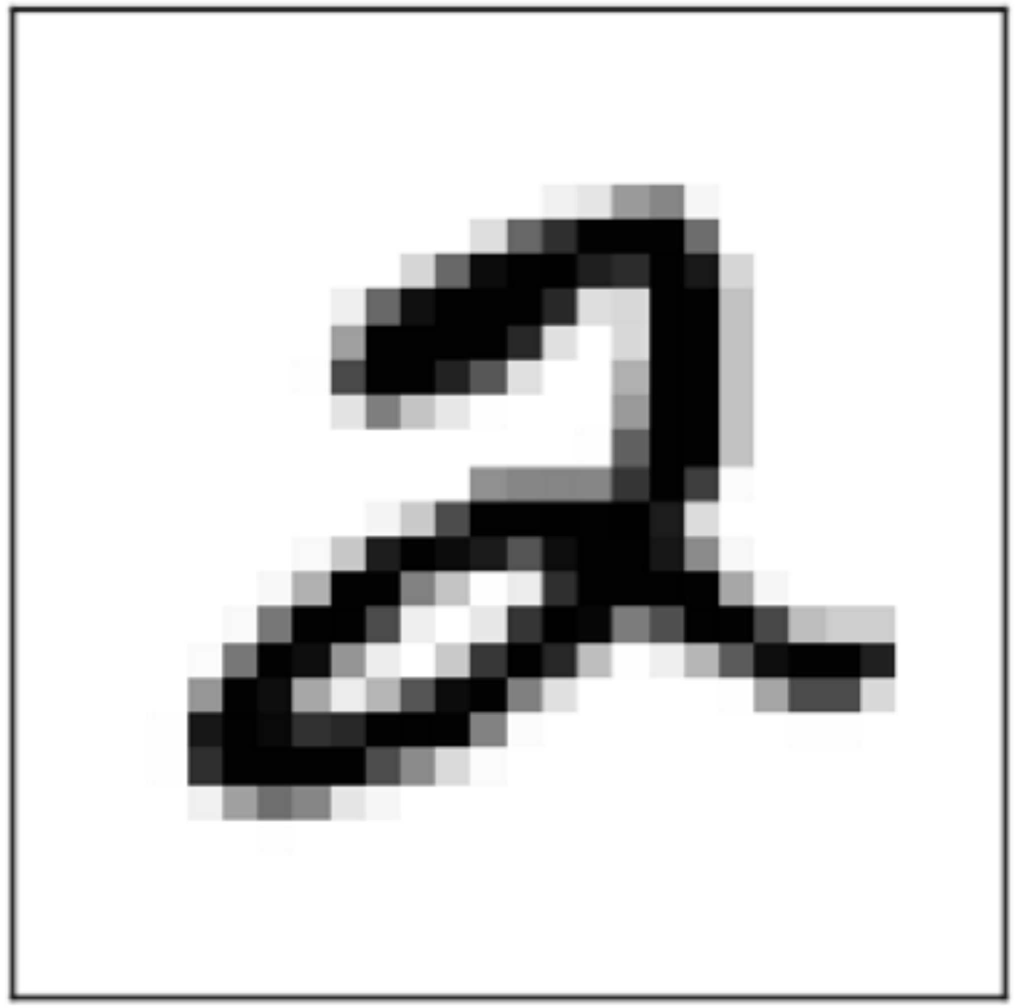


Word Vectors and the Word2Vec Algorithm

Matt Zhang

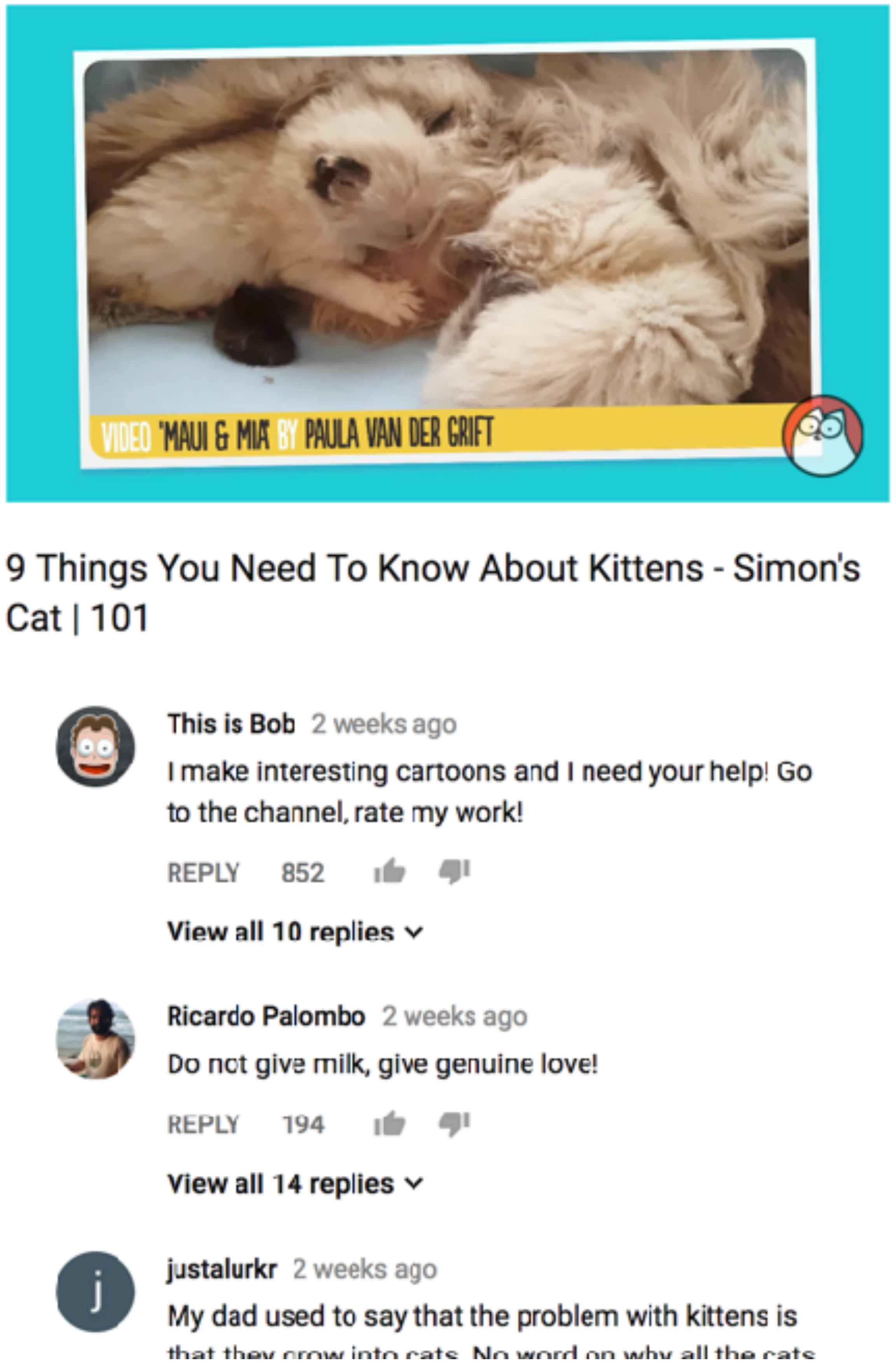
Word vectors are an interesting way of representing text. Similar to how we can treat every pixel of a picture as a dimension of an image vector, we can treat every word in a vocabulary as a dimension of a word vector.



N dimensional vector, where N = number of pixels

Lorem ipsum
dolor sit amet,
consectetur adipisicing
elit, sed do eiusmo tempor
incididunt ut labore et dolore magna aliqua.
Ut enim ad minim veniam, quis nostrud exercitation
ullamco laboris nisi ut aliquip ex ea commodo consequat.
Duis aute irure dolor in reprehenderit in voluptate
velit esse cillum dolore eu fugiat nulla pariatur.
Excepteur sint occaecat cupidatat non proident,
sunt in culpa qui officia deserunt mollit anim id est laborum.

N dimensional vector, where N = number of words in vocabulary



9 Things You Need To Know About Kittens - Simon's Cat | 101

This is Bob 2 weeks ago
I make interesting cartoons and I need your help! Go to the channel, rate my work!
REPLY 852
View all 10 replies

Ricardo Palombo 2 weeks ago
Do not give milk, give genuine love!
REPLY 194
View all 14 replies

justalurkr 2 weeks ago
My dad used to say that the problem with kittens is that they grow into cats. No word on why all the cats

Using just this idea, we can already begin implementing some naive vector logic. I will first show a simple application using a dataset of YouTube comments.

We take around 500 videos, with over 175k comments. The idea is that we take the comments of each video and turn it into the video's representation in word vector space.

First we normalize each video's comment word count by the total word count over all comments for all videos. This is so that common words like "the" don't vastly overcontribute to the results.

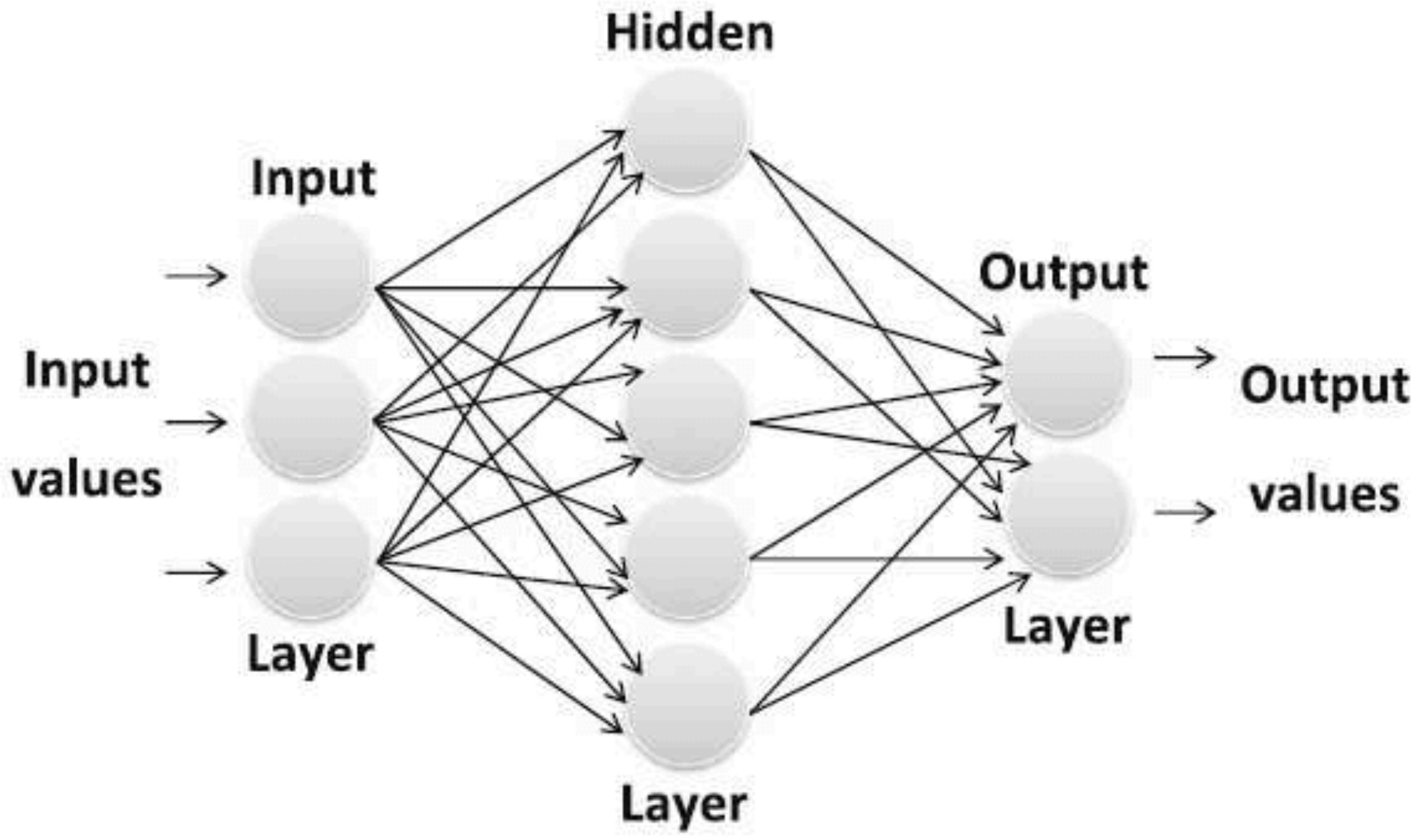
Then each word vector is normalized.

Finally, we define a distance between vectors as the usual L2 Euclidean distance. We can now compare videos by similarity of comments.

We can even add and subtract videos!

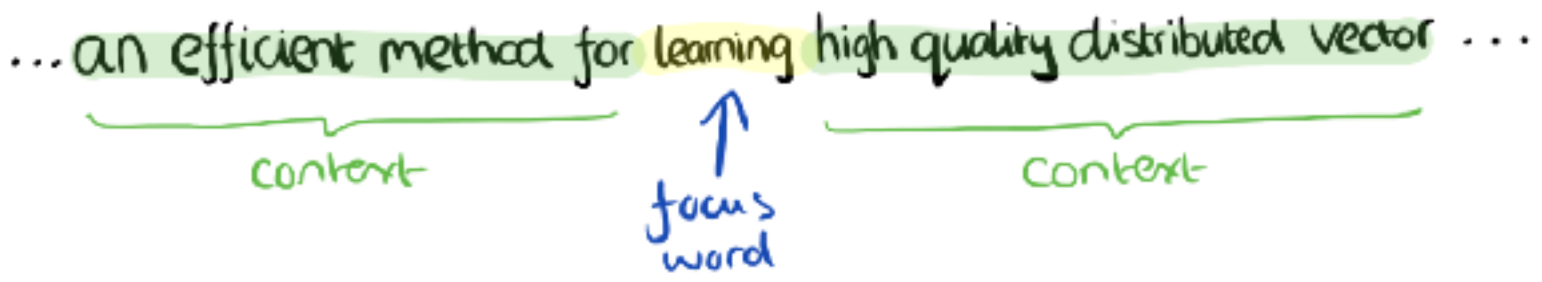
(Show first Jupyter notebook)

The results of the YouTube video analysis was somewhat interesting, but also contained a lot of trash. This is largely due to the fact that there were only around 500 videos in the dataset. However, another problem was the fact that the vocabulary space was very large.

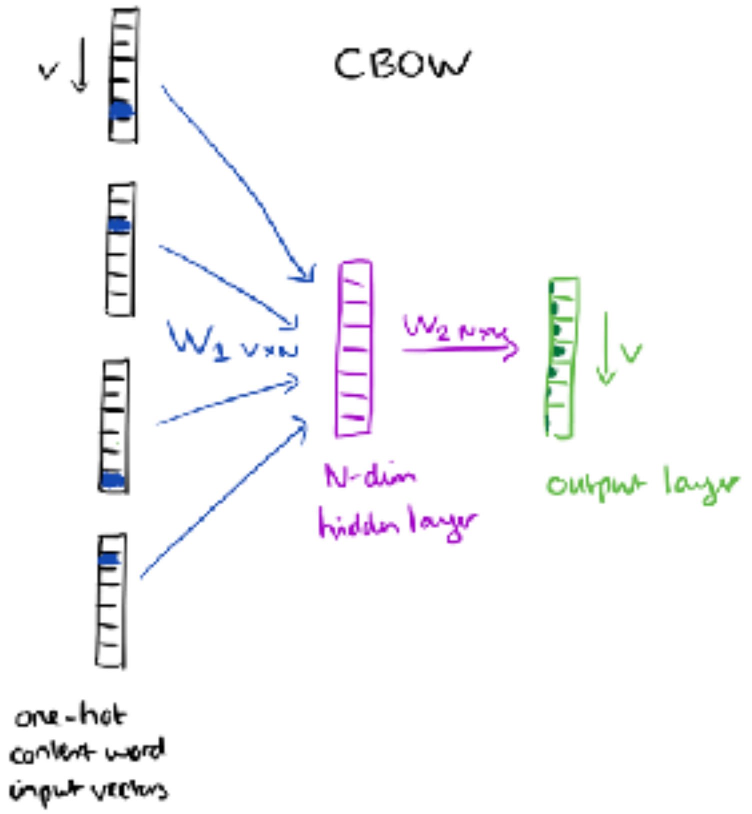


We can learn more about the connections between pieces of text by performing dimensionality reduction via a one-layer neural net.

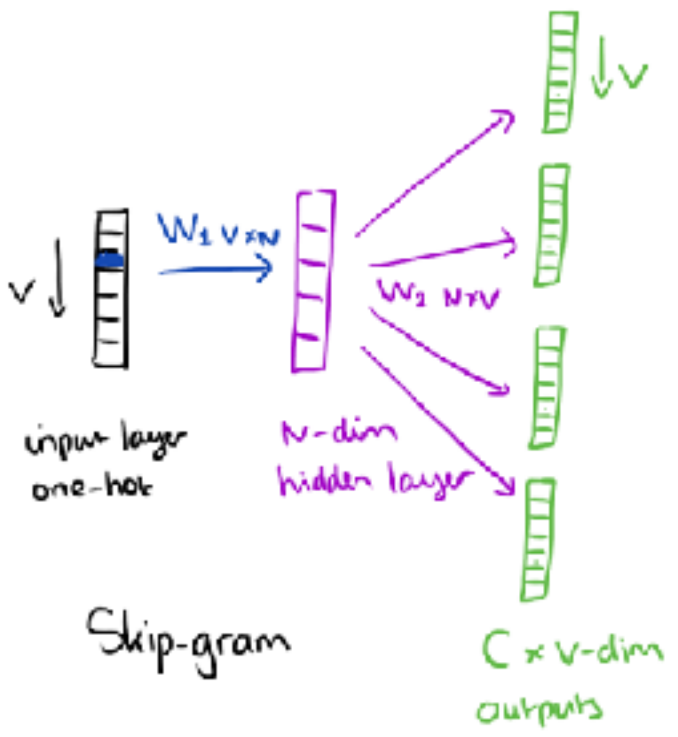
In a famous application called word2vec, we can find the relationship between words in an unsupervised way just by passing in a large passage of text. Taking each word in the text, we let the input and output of the neural net be the word itself and its eight surrounding words.



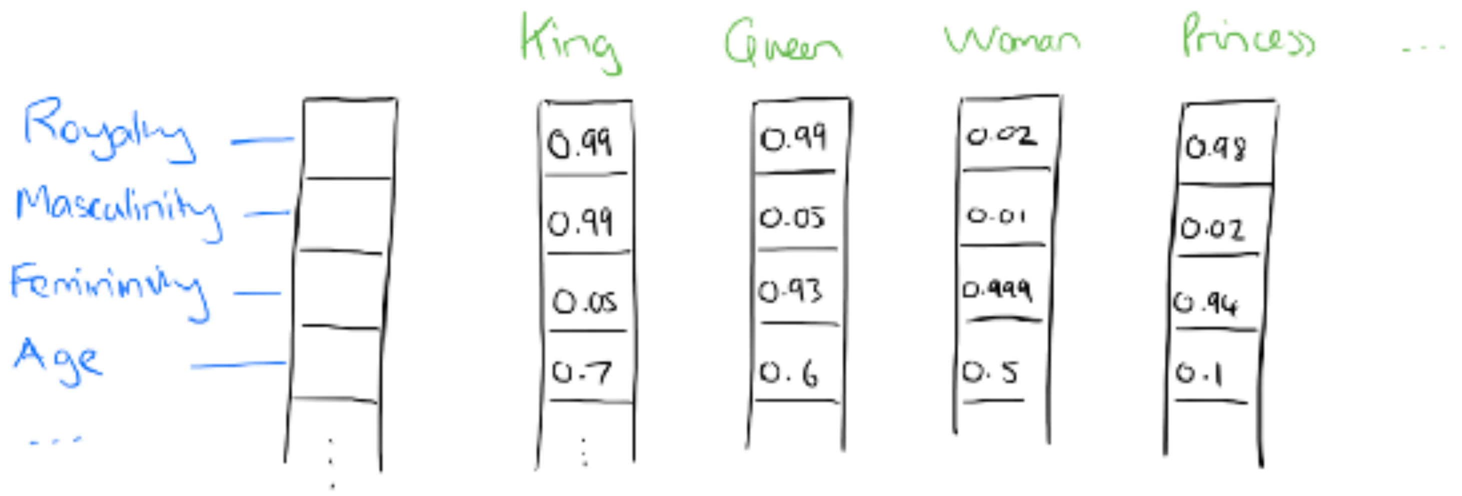
... an efficient method for learning high quality distributed vector ...
context
focus word
context



Either use focus word as output (continuous bag of words model) or use surrounding words as output (skip-gram model).



This training allows us to see interesting semantic connections between words. This allows us to then do word vector arithmetic, like the famous "king - man + woman = queen" example from "Linguistic Regularities in Continuous Space Word Representations by Mikolov et al. 2013".



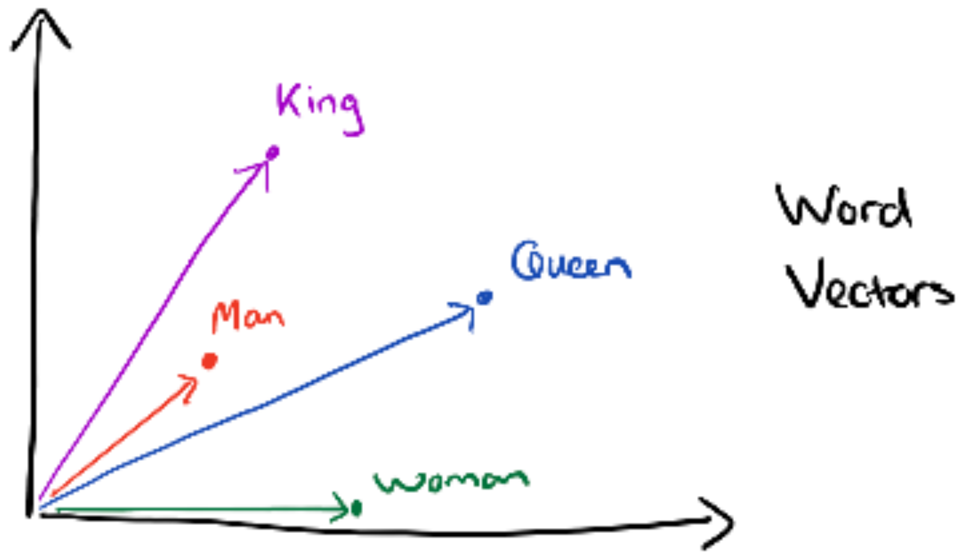


Table 8: *Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).*

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

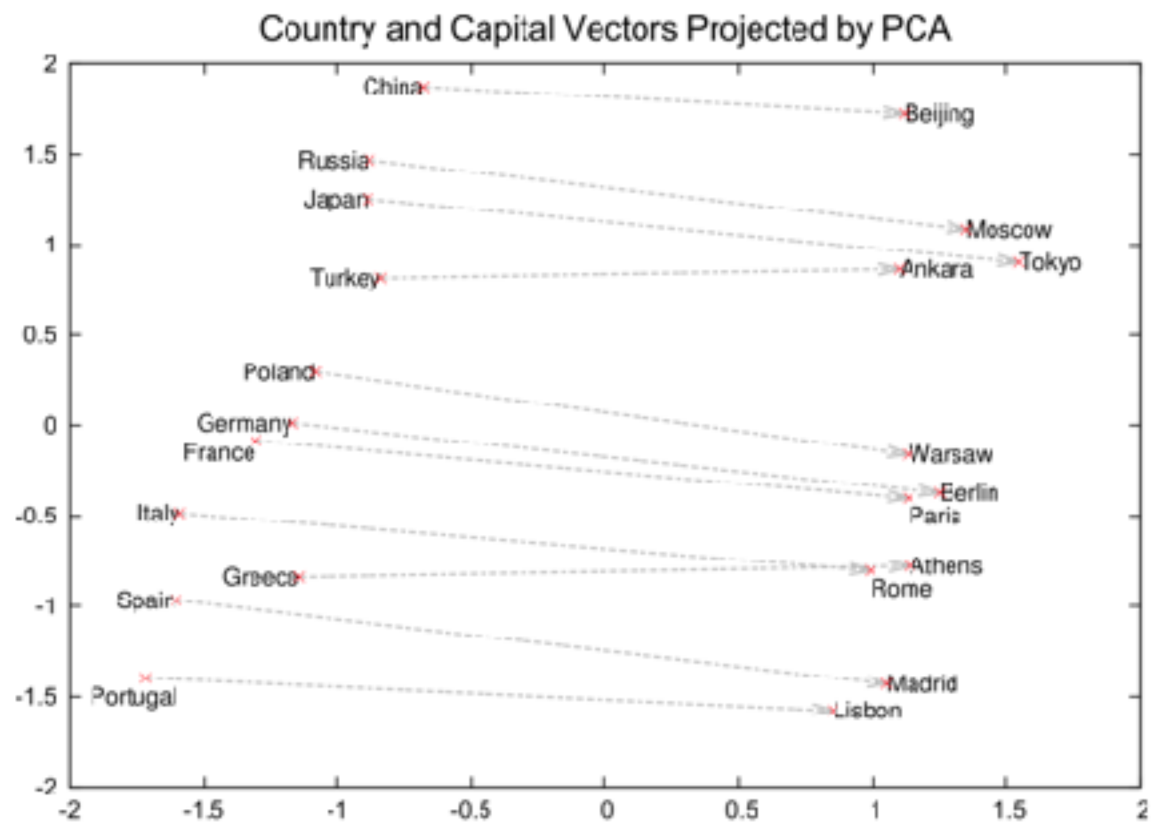
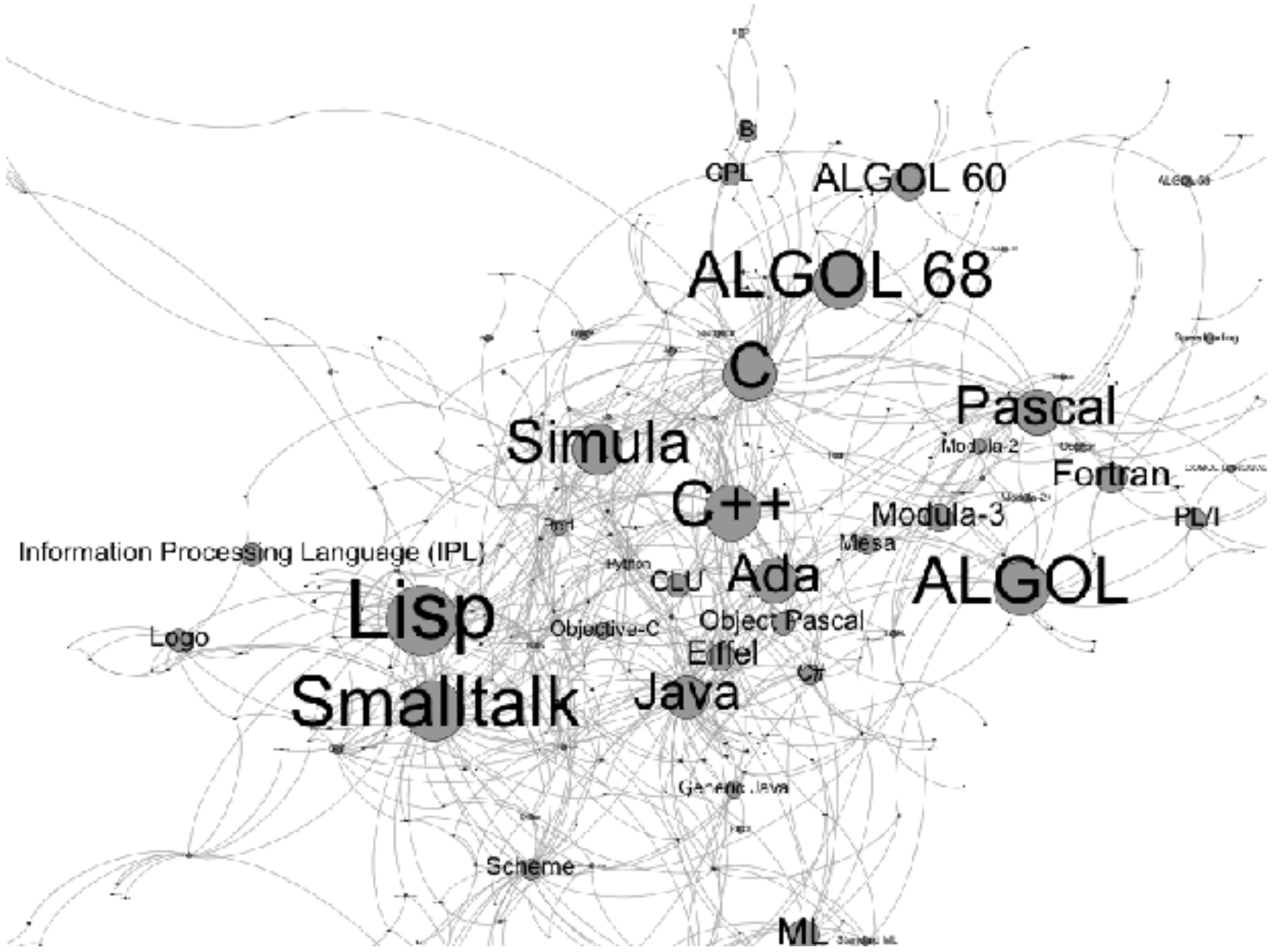


Figure 2: Two-dimensional PCA projection of the 1000-dimensional Skip-gram vectors of countries and their capital cities. The figure illustrates ability of the model to automatically organize concepts and learn implicitly the relationships between them, as during the training we did not provide any supervised information about what a capital city means.

Using the word2vec package in gensim, we feed in 1 million answers from StackExchange and see if our model can learn something about programming.



(Show second Jupyter notebook)

References

- [1] https://rare-technologies.com/word2vec-tutorial/ ➞ tutorial for using word2vec algorithm in gensim package
- [2] https://www.kaggle.com/datasnaek/youtube ➞ YouTube dataset, contains comments from top 200 trending videos each day
- [3] https://www.kaggle.com/stackoverflow/stacksample ➞ StackExchange dataset, contains lots of questions and answers.
- [3] https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/ ➞ blog post explaining word vectors
- [4] https://www.tensorflow.org/tutorials/word2vec ➞ explanation of word2vec algorithm
- [5] Linguistic Regularities in Continuous Space Word Representations – Mikolov et al. 2013 ➞ paper with “king - man + woman = queen” example