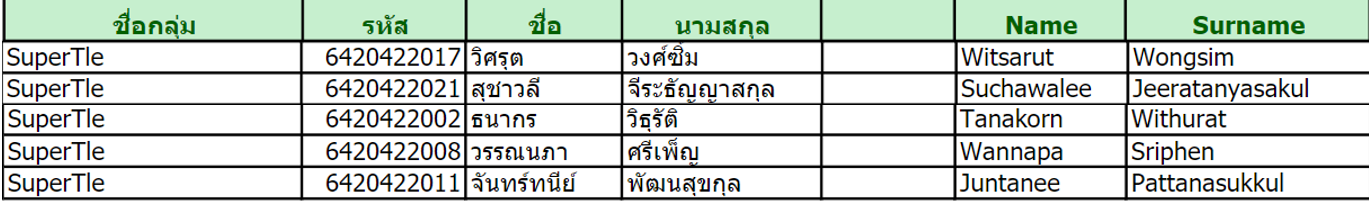
**Team Member**

****

**1. Implementing Word2Vec**

In this part you will implement the word2vec model and train your own word vectors with stochastic gradient descent (SGD). Numpy methods could be utilized to make your code both shorter and faster. The following requirements should be satisfied:

* a) Negative sampling loss
* b) Implement the skip-gram model from scratch
* c) Train with real-data
* d) Show the resulting embeddings

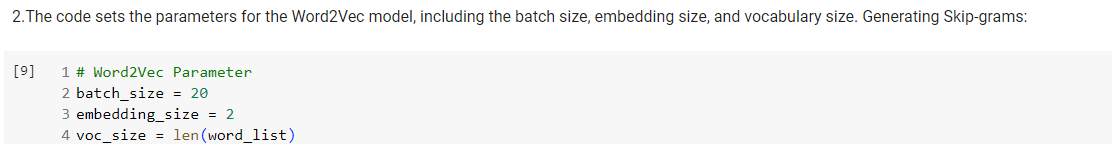
Numpy with negative sampling loss

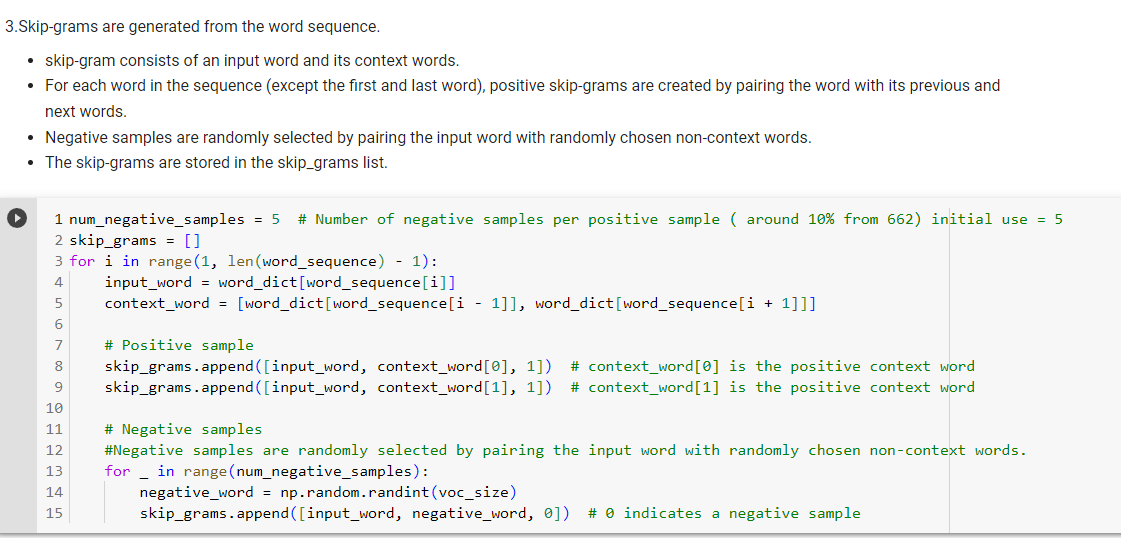
A picture containing text, algebra, receipt

Description automatically generated

A screenshot of a computer

Description automatically generated with low confidence





A picture containing text, receipt, algebra

Description automatically generated

A screenshot of a computer program

Description automatically generated with medium confidence

A picture containing text, screenshot, design

Description automatically generated

A picture containing text, screenshot

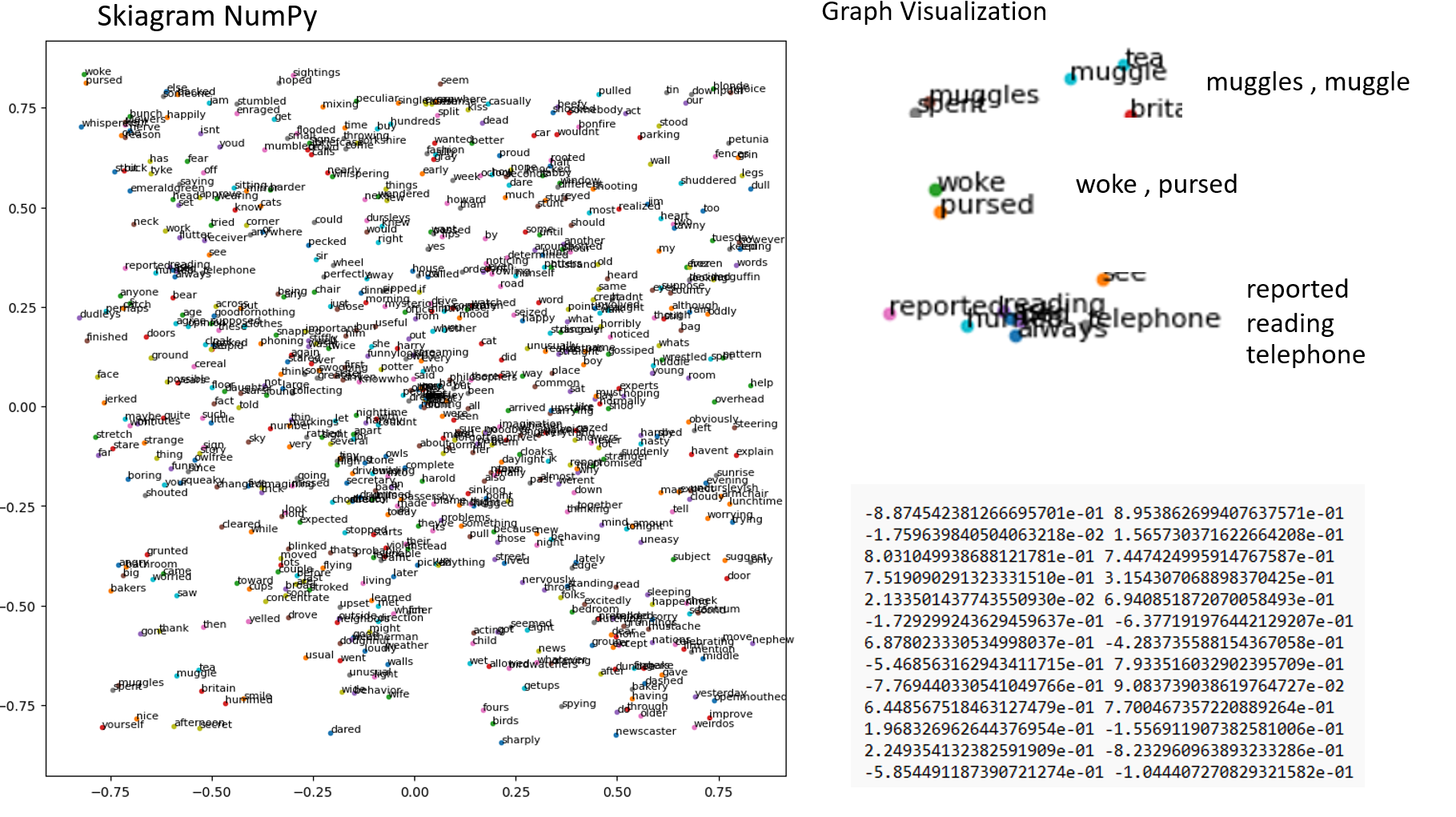
Description automatically generated

A screen shot of a computer screen

Description automatically generated with low confidence

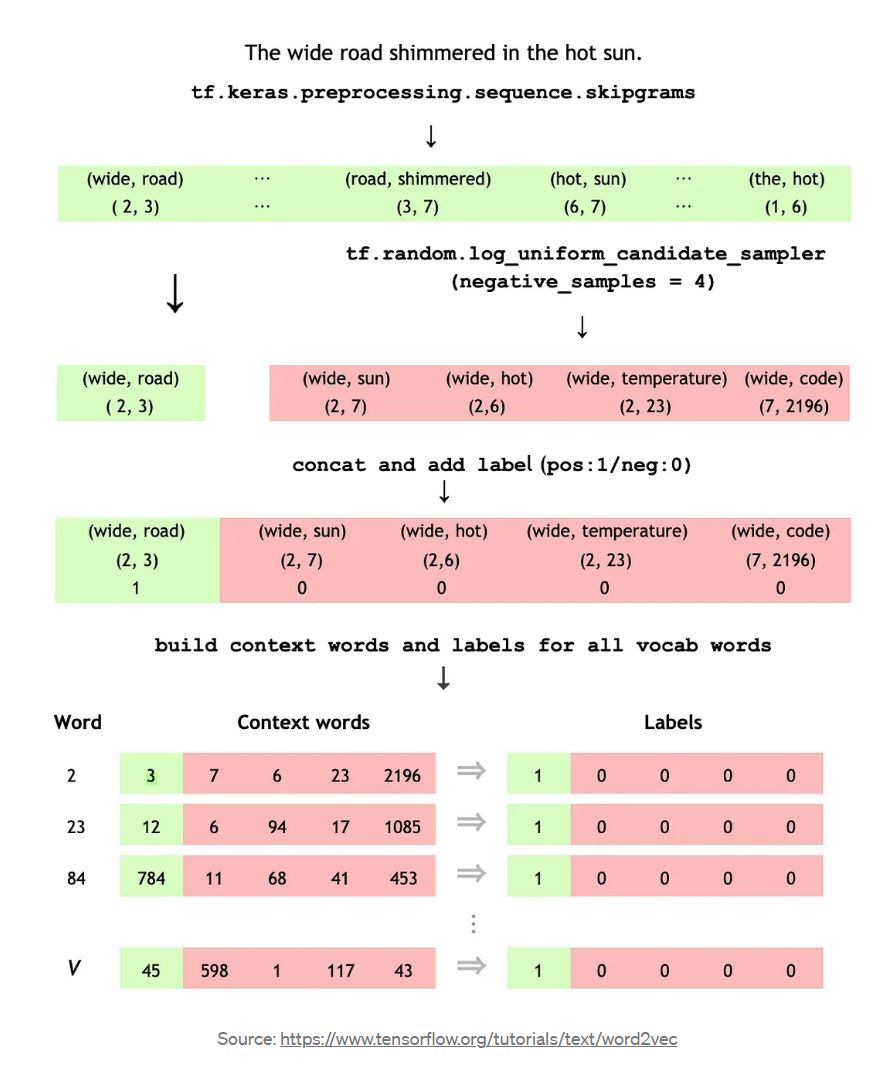
A picture containing text, plot, line, diagram

Description automatically generated

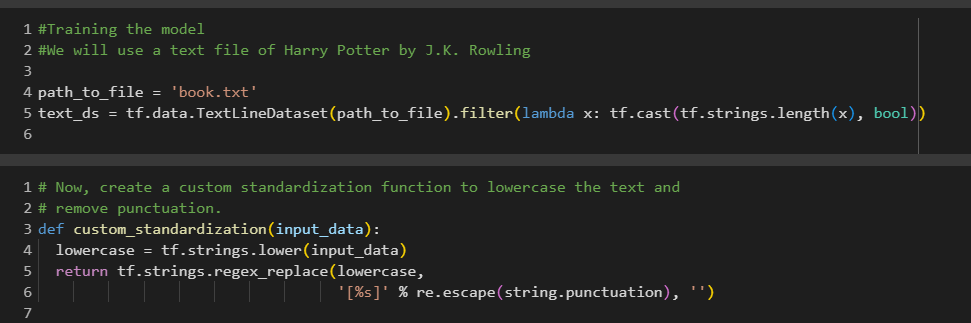


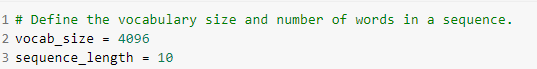
2.Tenserflow with negative sampling loss

Model Workflow

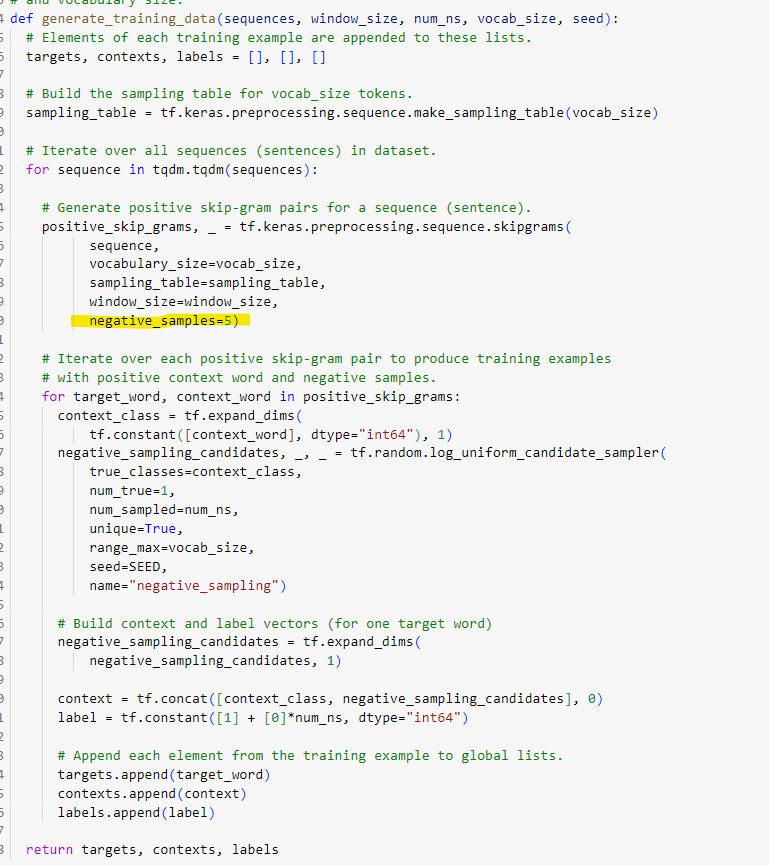


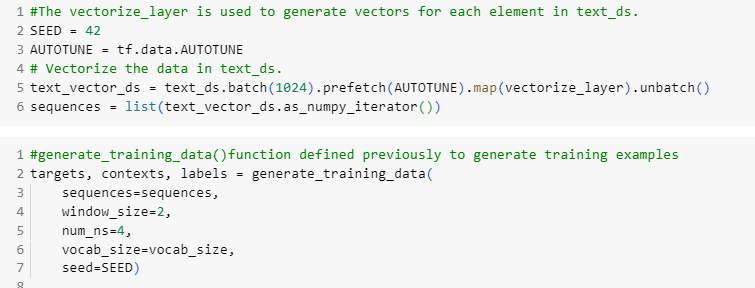
Get data and clean

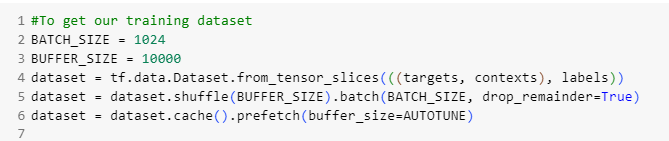


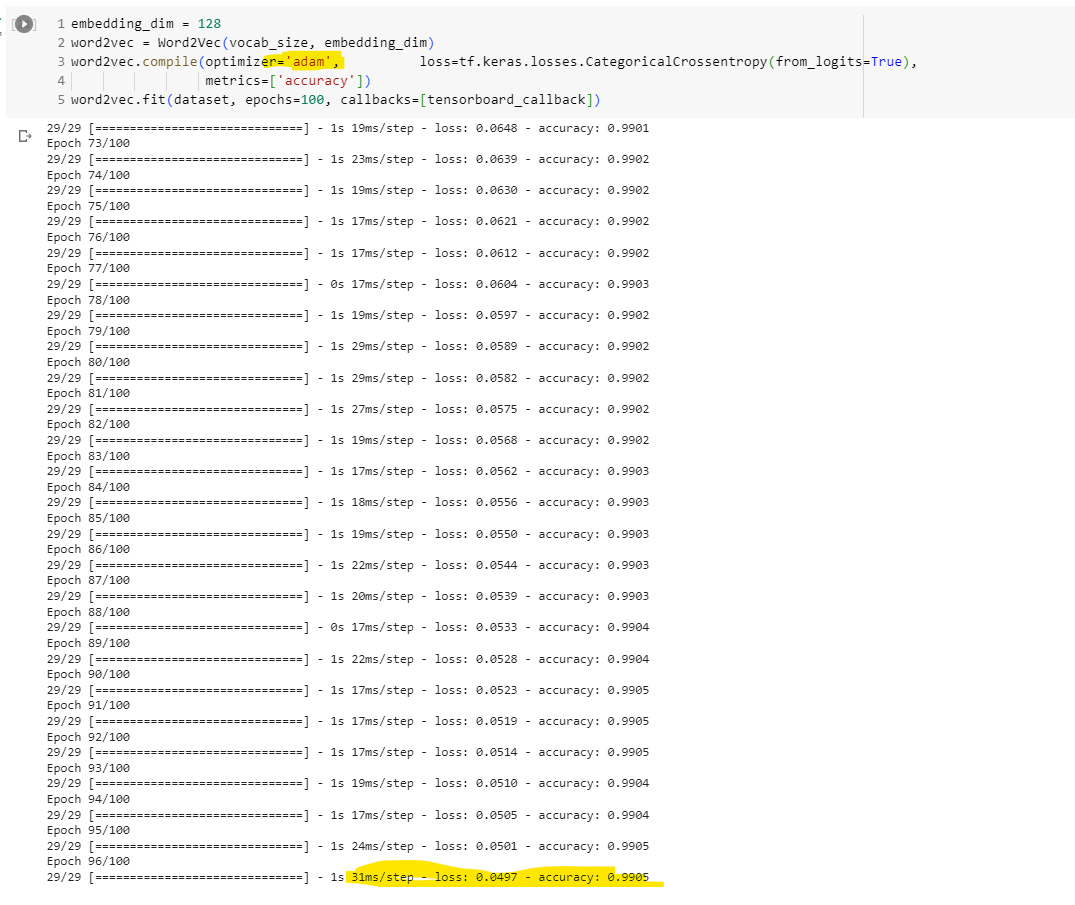




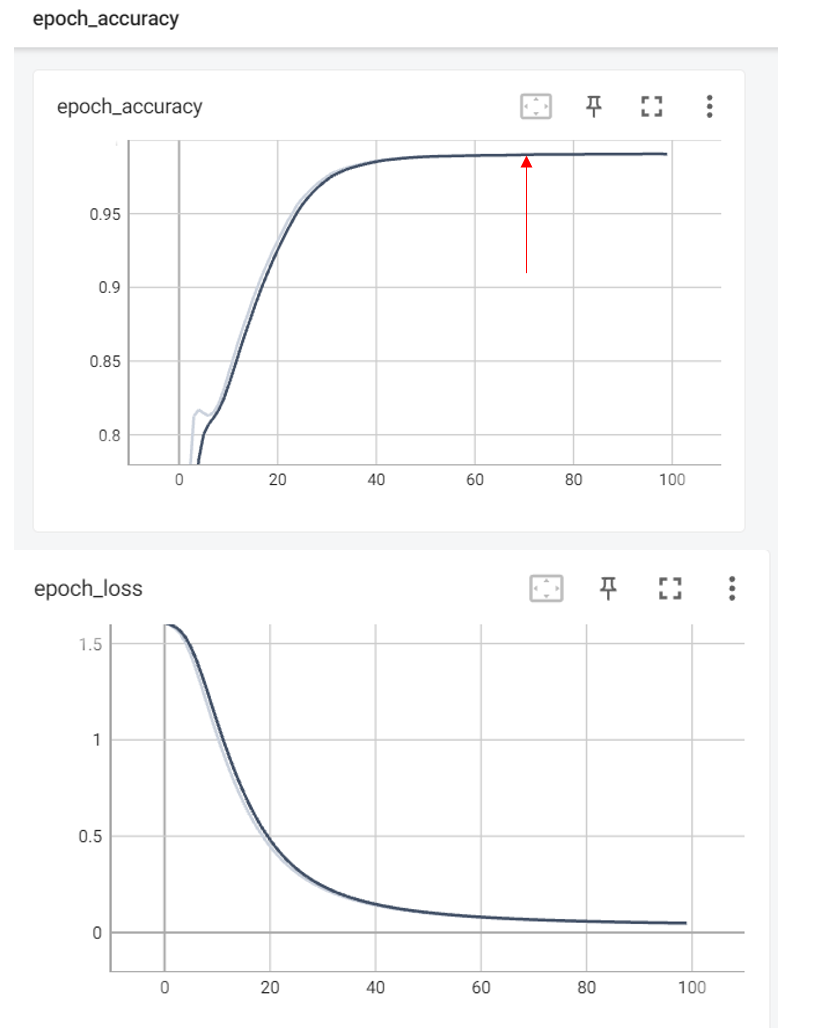


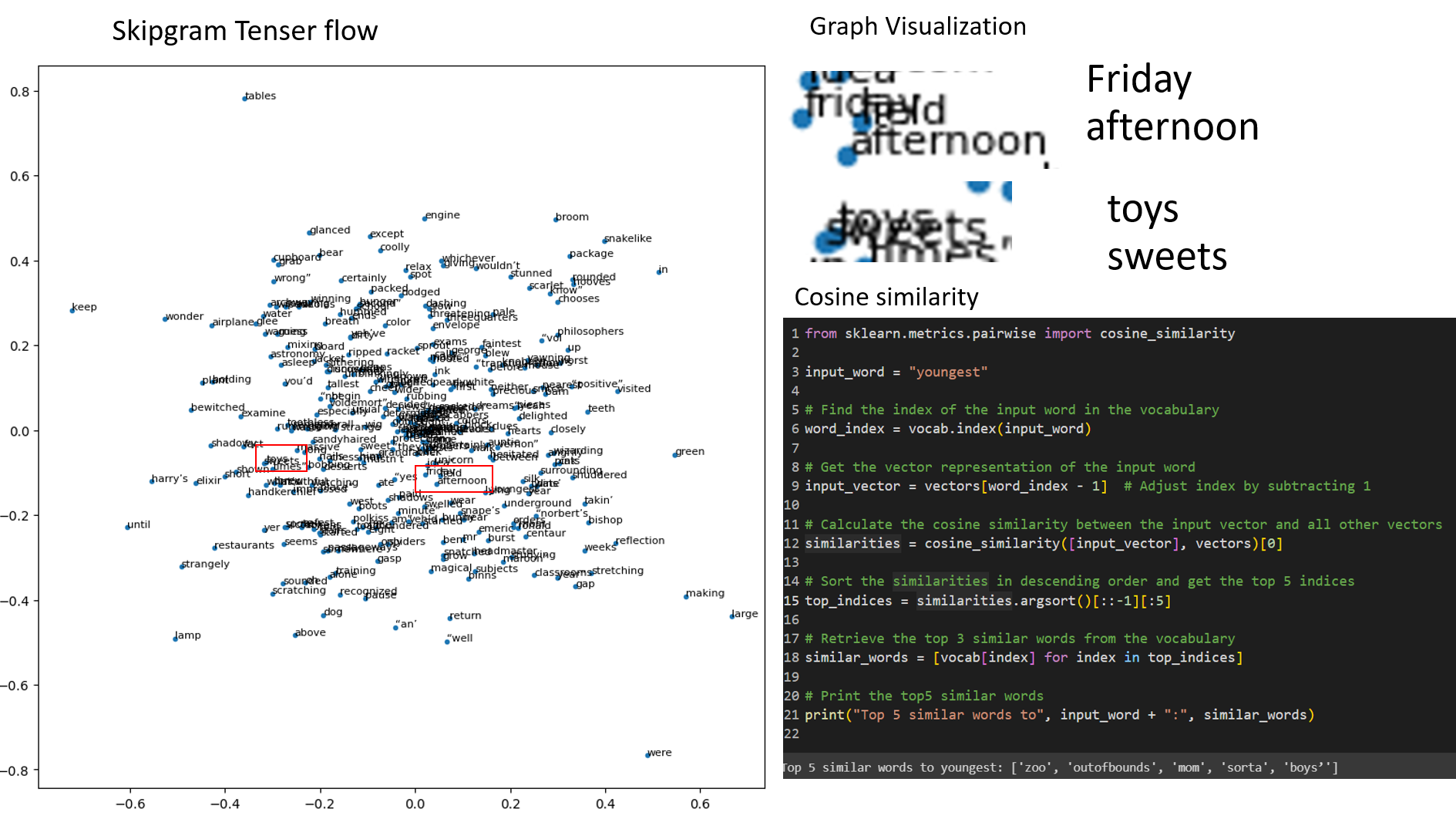






Epoch100 loss: 0.0482 - accuracy: 0.9904





input\_word = "youngest"

Top 5 similar words to youngest: ['zoo', 'outofbounds', 'mom', 'sorta', 'boys’']

3D plot

