Skip-Gram Model :

Tokenizing or one hot encoding is not a good representation of words as it misses the semantic meaning of the words. For example, say you 1k words in the vocabulary and the input sentence is orange is juice, apple is juice. Say Apple’s token is 750 while orange’s token is 230. Therefore you cannot use any distance based methods to determine similarity that apple and orange are similar. We need a more robust way of representing words such that similar words have similar vectors. This is done using word embedding algorithms

Word2vec or skip gram model :

Say we have input sentence, “ I want a glass of orange juice to go along with my cereal “

Now to capture semantic information through context we make context, target pairs where target is randomly chosen within a window say + or - 3

Context = orange, orange, orange Target = juice, want, along

Now we build a model to map context to the target with vocab = 1k and embedding dim = 300

Input = tokenized word = say tokenize(orange) = 1k dimensional vector with 450th entry =1 and rest = 0. Input = (num\_of\_ex, 1k)

Now we have a hidden layer call it E with 300 units. Weights are randomly initialized. We multiply input ( tokenized words) with E matix to get output of ( num\_of\_ex, 300).

Now we multiply E matrix with output layer weights (300\* 1k) and do softmax activation to get probabilities for all 1k words (num\_of\_ex, 1k). We use gradient descent to find right weights for E matix and output weights such that the probabilities of 1k words match the target probabilities ( having highest for target word)

Therefore, the trained 300 dimensional weight matrix is our word embedding.

Dimensions = ( say we have 10k examples, 1k vocal size, 300 embedding dimension)

(1k , 10k) \* (10k, 300) = (1k, 300k)

(1k, 300k) \* ( 300k, 10k) = (1k, 10k)

Negative Sampling

Skip gram is expensive to compute because of the softmax denominator term which sums over 10k vocab.

Here instead of having context as input and target as label, we have context, target as input and label is Yes/No indicating weather context, target pair are semantically related. We construct dataset as follows,

For every positive example ( C,T pairs within the window) choose a negative example. Say k = 3

Orange - juice

Orange - king

Orange - queen

Orange - school.

Now the model is the same as above. Input -> E -> output. This is less expensive as we dont run on entire 10k examples but instead we run on subset, i.e, k+1 ( k negative, 1 positive)

Glove :