**Word Embedding:**

There are two ways to do word embedding CBOW and Skipgram

**Skipgram:**

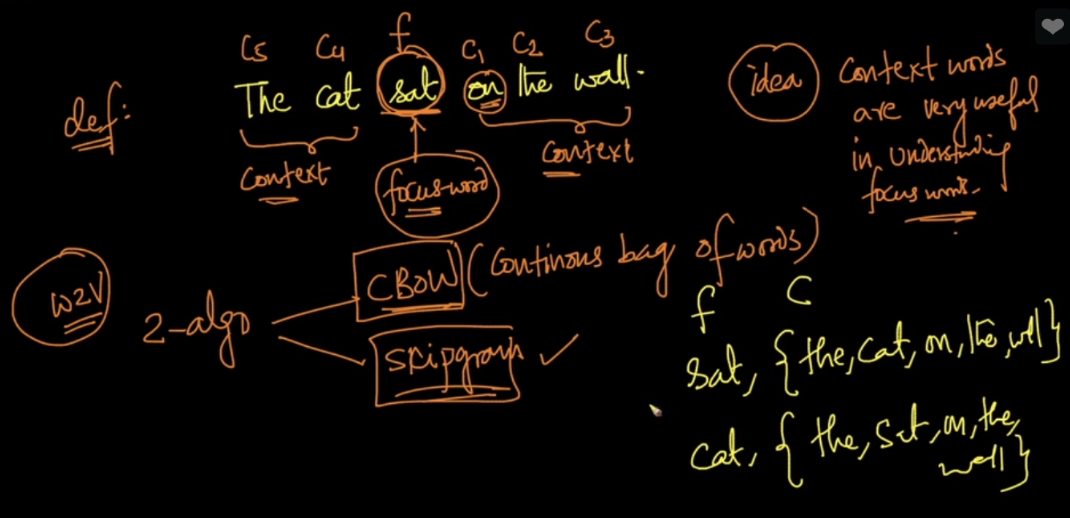
**Context and focus-words:** As shown in below image **sat** is a focus word and other words are context words, so the idea is we’ll use context words to find focus words.

Example: A glass of orange ………..

So the output word should be juice, and how we find we provided all 4 context words and our model gives juice as output.

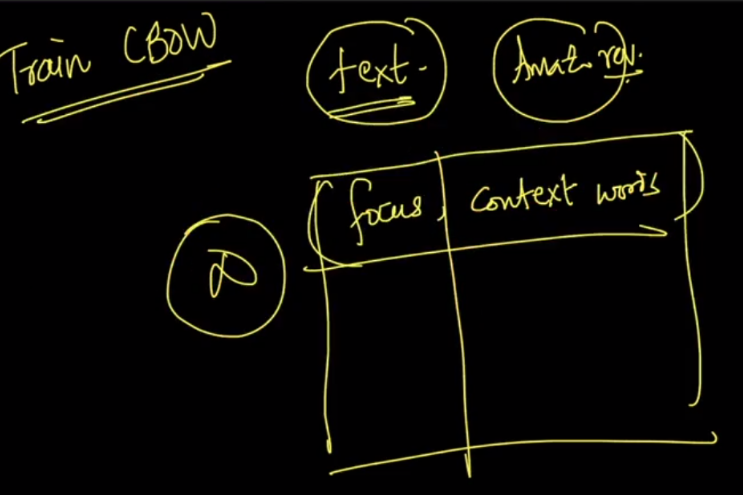
It’s not necessary to keep all the remaining words as context, we can keep 1-1 word on left & right, Example for **sat**, context words are **cat, on**.

We can have 4 words on left, Ex: A glass of orange **juice**. So for **juice** we have **A, glass, of, orange** as context words



So what we do is create a focus and context words dataset from given corpus.

So focus column contains focus words and context column contains list of context words.

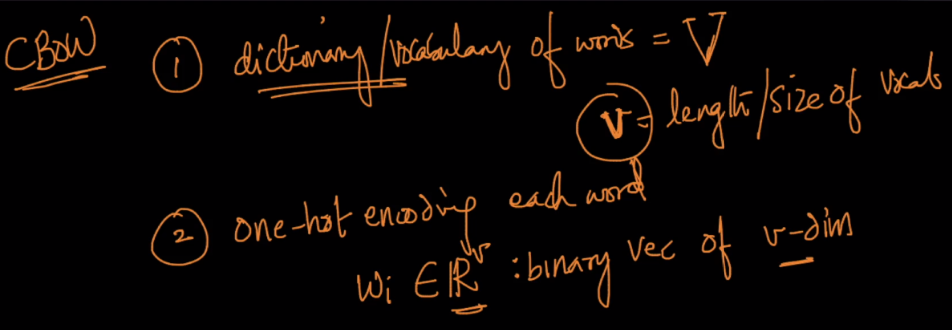


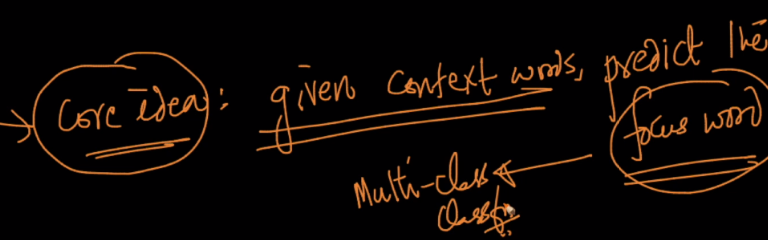
Now we create a dictionary or vocabulary of unique words present in whole corpus, let’s say in whole corpus there are v unique words, so size of vocab will be v.

Now for each word we have one hot encoded form which is of v dimension.

Suppose we have 10k unique words in corpus and therefore v=10k.

Now new word, say ‘juice’ and it’s present at 150th index, so the vector for ‘juice’ will have 10k dimension with 1 at only 150th position and rest will be 0.





So we’ll find the vectors for all provided context words.

**Architecture of skip gram:**

**Input Layer:** fixed no. of context words’ vector each of ‘v’ dimension.

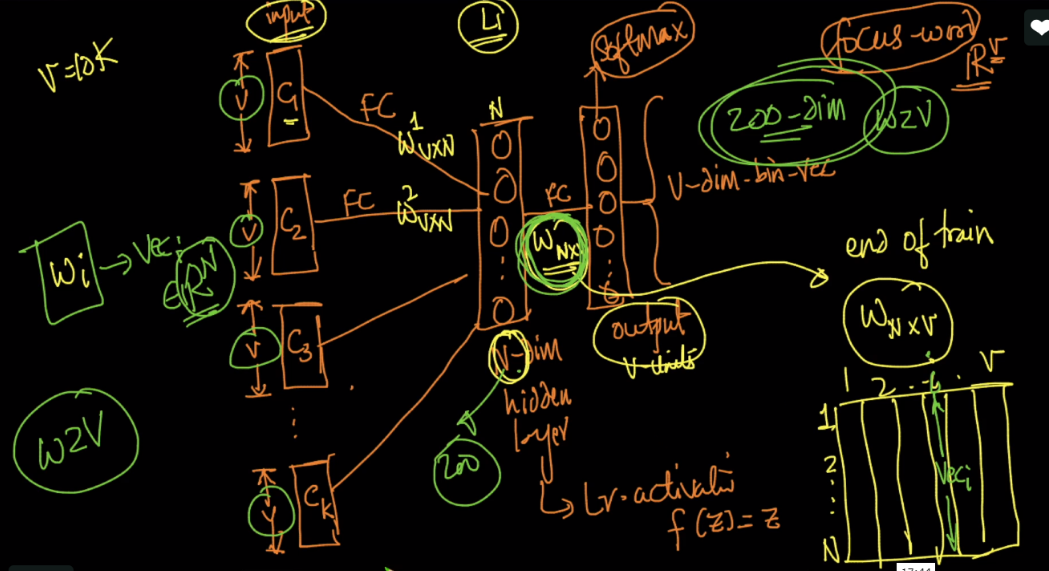
**Hidden Layer:** It’s of N dimension. And it has linear activation function { f(z) = z} in each unit. And this layer dimension specifies the dimension we’ll get for the focus words (ex for w2v we get 300-dim vector for each word, that means w2v has 300 units/dim in this hidden layer)

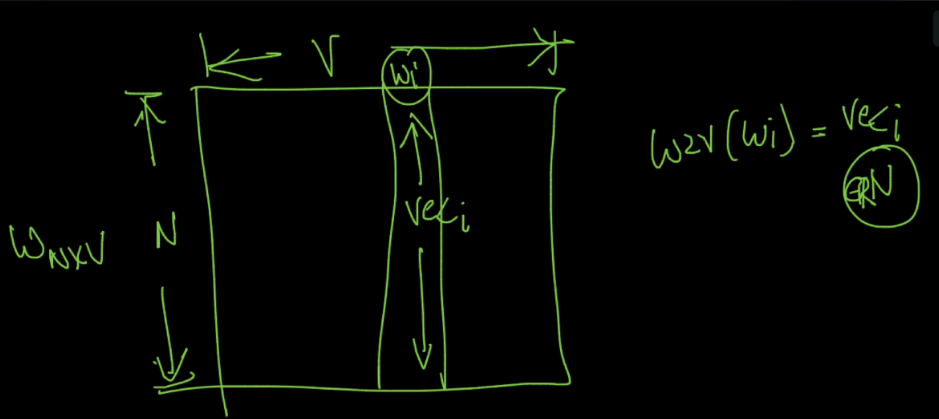
**Output layer:** It’s a softmax classifier layer which will have v-units one for each word in vocab, and outputs the v-dimensional vector. Let’s say input is “a glass of orange” and output will “juice”, and if ‘juice’ is at 150th position so it will output v-dim vector in which only 150th position have 1 and rest will be 0.

**Note:** it’s a full connected network.

From hidden to output layer there will be n \* v links, that means there are n\*v weights.

Now if we want to find the word embedding of ‘juice’ word we make table of weights present between hidden to o/p layer which have ‘n’ rows and ‘v’ columns, And since ‘juice’ is at 150th position in vocab, so 150th column will be the word embedding of ‘juice’.





Implementing CBOW: <https://www.kdnuggets.com/2018/04/implementing-deep-learning-methods-feature-engineering-text-data-cbow.html>