**Gender Debiasing Algorithm:**

In this project, we aim to perform hard gender debiasing on the pre-trained Glove [1] embedding. There are several versions of the pre-trained embedding available. For this project, we have chosen the 50-dimensional version, which is based on Wikipedia 2014 and Gigaword5 and has 400,000 words.

The gender debiasing algorithm has the following four **input parameters**:

1. The embedding vectors that we aim to debias. In this case these are the 50-dimensional Glove embedding vectors
2. A list of gender specific words which should not be debiased. These are words like *man, woman, boy, girl* etc. In our case, we read the list from the **gender\_specific\_seed.json** file which contains **218** gender specific words.
3. A list of gender definitional word pairs (tuples) which is used to calculate the gender subspace. In other words, these are the words that will define the gender direction in the embedding space. Examples of these word pairs include *(woman, man), (girl, boy)* etc. In our case, we read these pairs from the **definitional\_pairs.json** file, which has **10** such word pairs.
4. A list of equalize word pairs (tuples). The goal is that after the debiasing, any word that is not gender specific should be at equal distance from both words in each of these pairs. Examples of these word pairs include *(woman, man), (girl, boy)* etc. In our case, we read these pairs from the **equalize\_pairs.json** file, which has **52** such word pairs.

It is important to note the distinction among parameters 2, 3 and 4 above. The words in all 3 parameters are essentially gender specific. However, parameter 2 is a discrete list of words not to be debiased. Parameter 3 is list of *pair* of words for defining the gender subspace and parameter 4 is also a list of *pair* of words for equalizing purpose. Theoretically, parameters 2 and 3 could well be the same list. However, in our case parameter 3 is a superset of parameter 2.

The output of the algorithm is the debiased embedding vectors.

Now that we know what the inputs and output are, let’s dig into the algorithm itself. The algorithm at a high level has two major steps:

1. Identifying the gender subspace
2. Neutralizing and Equalizing the words

We should point out that we perform the algorithm on a normalized version of the embedding, so that all vectors in the embedding are unit vectors. We are primarily interested in the bias direction and the cosine similarities between vectors, and not the exact magnitudes. So normalization does not hurt.

Let’s look at the details of how the algorithm steps work.

**Identifying the gender subspace:**

Mathematically, the gender subspace is ***B*** is defined as a set of ***k*** orthogonal unit vectors {*b1, b2,...,bk*} where ***k*** is the number of components we want in the subspace. In our case, we take ***k*** =10. Also, since we are using the 50-dimensional embedding vectors, the individual unit vectors in the subspace, namely *b1, b2,...,bk* will all be 50-dimensional vectors as well. To obtain the gender subspace, we follow the following steps:

1. Define an empty list **L**
2. For each pair vector (***x, y***) in the gender definitional pairs:

* Compute the center vector ***c*** as
* ***c = (x + y) / 2***
* Calculate difference vectors ***diffx = c – x*** and ***diffy = c – y***
* Append ***diffx*** and ***diffy*** to the listL.

1. Apply Principal Component Analysis (PCA) on the list **L** with 10 components. The set of 10 components {*b1, b2,...,b10*} is the gender subspace

**Neutralizing and Equalizing the words:**

Once we have the gender subspace, we are left with the task with neutralizing and equalizing words. This is done in a two-fold approach.

1. For each word, we calculate the projection of that word onto the gender subspace. If the word is not gender specific, we subtract the projection from that word, so that the resulting vector is effectively the projection on the orthogonal subspace of the gender subspace.

The projection of a word vector ***w*** onto the gender subspace {*b1, b2,...,b10*} would be:

So if ***w*** is not gender specific, we update vector ***w*** as***:***

Note that a normalization is performed after the subtraction. For implementation convenience and performance, we first calculate the subtraction results and perform a normalization over all vectors of the embedding.

1. The final task is to make sure that any word that is not gender specific is at equal distance from both words in each of the equalizing pairs. For doing this, we perform the following steps for each pair vector (***x, y***) in the equalizing pairs:

* We calculate the mean vector of the pair:
* We calculate the projection of this mean vector onto the gender subspace using the same subspace formula as before:
* We subtract the projection from the mean vector and calculate a vector ***v:***
* For both words ***x*** and ***y***, we update their corresponding embedding vector as follows:

Note that ***xB***and ***yB*** have already been calculated in step 1 above.

Now that the vectors have been updated one more time, we perform one final normalization of all the vectors.

In the accompanying notebook, we have read in a set of words denoting different professions from the **professions.json** file. This set is an ideal example of words that are not gender specific. We havedemonstrated that after the equalizing step is done, in the resulting debiased embedding, each word in this set is at equal distance from both words in each of the equalizing pairs, which is not the case in the original embedding.