- The 2 notebooks have the updated code which i will describe in a bit.
- The 2 text files have more keywords for drought, earthquake, flood and disaster which we have extracted from the chinese room editor and used to increase our precision and improve our modelling.

Updates:

- We have added a performance metric to our modelling. The code generates the confusion matrix for the models. This is used to calculate the precision, recall and thereafter the F score. **Lines 137-140 & 210-212**

- This performance measure was used to calculate the F score for Oromo with the following seeded words: **Lines 130-140**

```
In [130]: # Anchor words
          #["chocho'a", "lafaa","socho'a","lafa"],

          #socho‰aa socho‰an socho‰uun  socho‰uu
          anchor_words = [["hoongee","hongee"],["galaana","galaanaa"],["balaa"]]

          anchored_topic_model = Corex(n_hidden=25, max_iter=1500,seed=3192)
          anchored_topic_model.fit(doc_word_mat, words=words, anchors=anchor_words, anchor_strength=6);
```

and it returned the following:

Out[137]:

	Predicted++	Predicted--
Actual++	161	1645
Actual--	20	144

```
In [138]: precision = cm.loc["Actual++","Predicted++"]/(cm.loc["Actual++","Predicted++"] + cm.loc["Actual--","Predicted++"])
          precision
Out[138]: 0.8895027624309392

In [139]: recall =  cm.loc["Actual++","Predicted++"]/(cm.loc["Actual++","Predicted++"] + cm.loc["Actual++","Predicted--"] )
          recall
Out[139]: 0.08914728682170543

In [140]: F1_score = 2*(precision * recall)/(precision + recall)
          F1_score
Out[140]: 0.16205334675390037
```

- Now we used the words extracted from the chinese room editor and added them to the seeded words as follow: **Lines 140-212**

```
In [141]: # Anchor words
          #["chocho'a",  "lafaa","socho'a","lafa"],

          #sochoʔaa sochoʔan sochoʔuun  sochoʔuu
          anchor_words = [["hoongee","hongee","oolaa"],["galaana","galaanaa","guutuu","lolaa"],
                          ["balaa","dhibee","irraa"],
                          ["sochii","lafaa"]]

          anchored_topic_model = Corex(n_hidden=25, max_iter=1500,seed=3192)
          anchored_topic_model.fit(doc_word_mat, words=words, anchors=anchor_words, anchor_strength=6);
```

and it returned an improved precision and recall score of:

```
In [210]: precision = cm.loc["Actual++","Predicted++"]/(cm.loc["Actual++","Predicted++"] + cm.loc["Actual--","Predicted++"])
          precision
Out[210]: 0.9018181818181819

In [211]: recall =  cm.loc["Actual++","Predicted++"]/(cm.loc["Actual++","Predicted++"] + cm.loc["Actual++","Predicted--"] )
          recall
Out[211]: 0.13732004429678848

In [212]: F1_score = 2*(precision * recall)/(precision + recall)
          F1_score
Out[212]: 0.2383469485824123
```

- This shows that once we retrieve the topic model with limited seeded keywords and then follow it up by searching for more keywords we are bound to get a better score. A more precise model.
- Following this, **Lines 219 to 224** have the Top 10 documents extracted according to the Occurrence Count of the seeded word for a given topic. This can also be used to extract top 20-25 documents for the same but the count reduces to 1 after the top 10 documents.

Issues:

- We were planning to run the same thing for Tigrinya but the chinese room editor returned the romanized version of the Tigrinya text. Therefore we tried to trans-literate it online but we were not able to get the correct text.
- We are meeting Ulf on Friday to correct this issue and also understand more about the CRE so that we can try and incorporate some of the methods automatically in the system.

Looking Forward:

- Given that we have a performance measure : precision, recall, f score, top 10 occurrence, specificity and sensitivity. We are going to expand this to Guided LDA seeding and compare the results.
- We are also looking forward to creating our own topic modelling system and compare the scores for the same.

Please review the document along with the notebook and give us your feedback on the same.