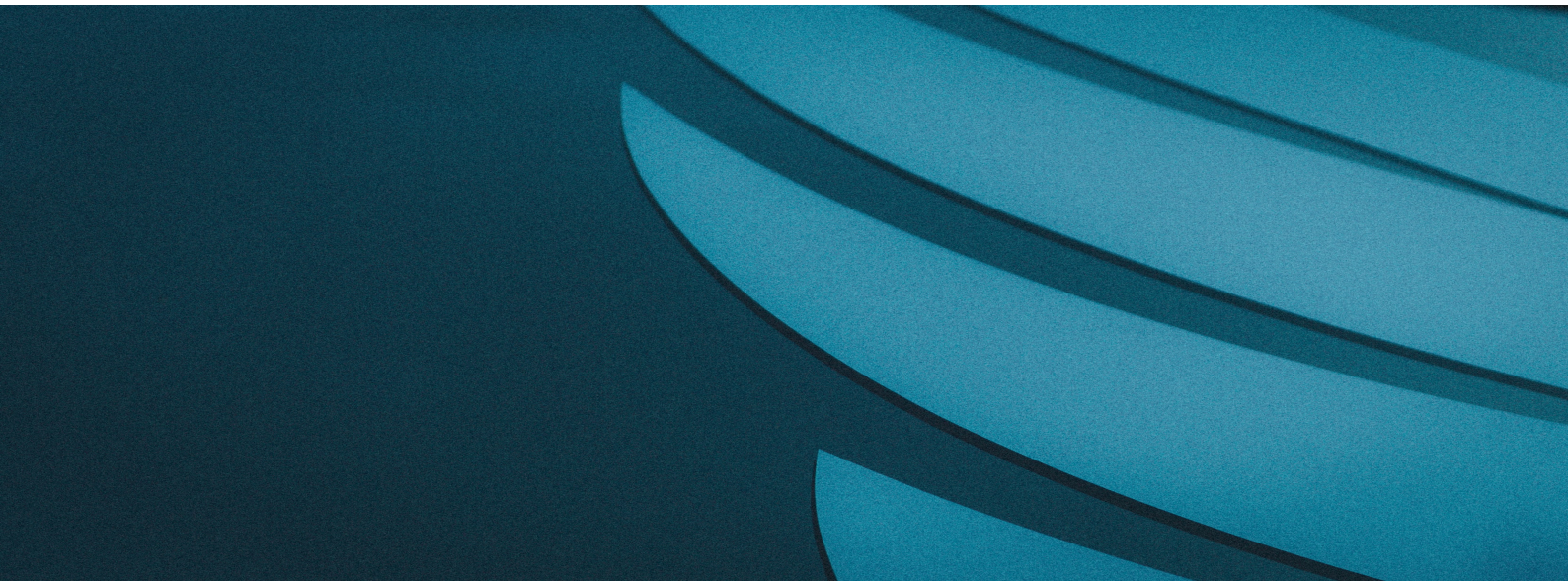
# What the Tweet?

## DETECTING ARABIC-LANGUAGE POLITICAL MISINFORMATION ON TWITTER



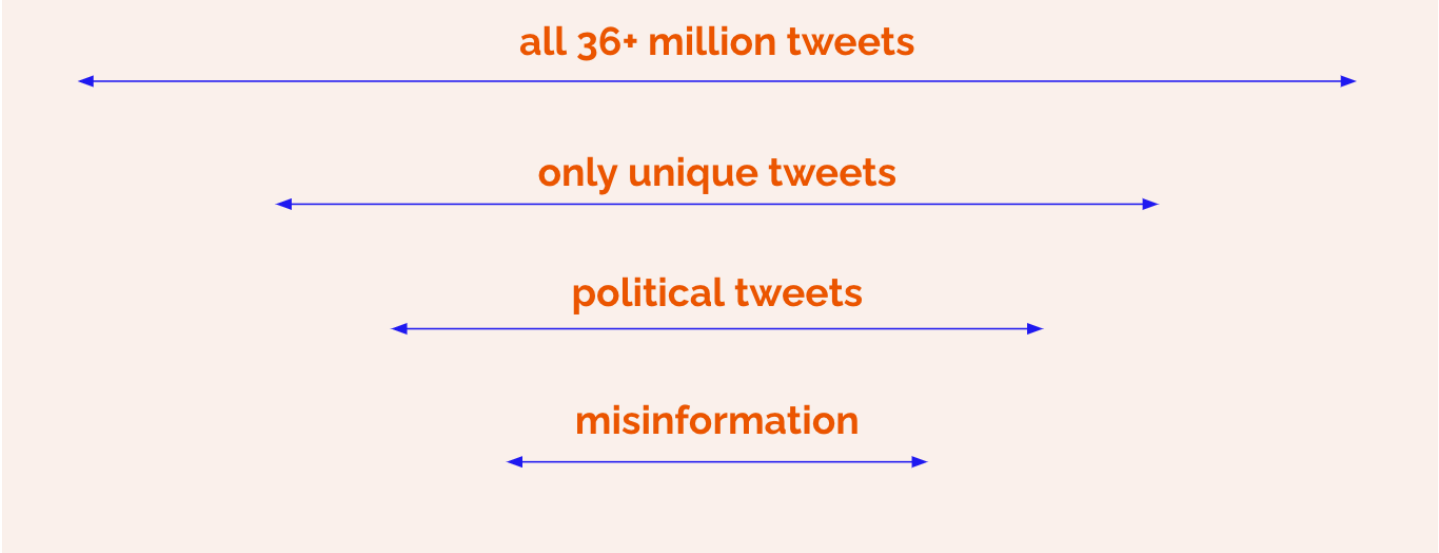
Detecting (political) misinformation and hate speech is a major challenge for organisations in the political, humanitarian and intelligence sectors. This project aims to contribute to the ongoing research in this field by looking specifically at the tools and processes necessary to detect political misinformation in Arabic-language tweets.

The project makes use of a dataset from the Twitter Transparency Center consisting of 5,350 Twitter accounts (and all of their 36+ million tweets) that have been identified by Twitter as being part of a state-linked Information Operations.

The project is motivated by the urgency described above, as well as a personal fascination for the Arabic language and a curiosity to discover the potentials for Arabic NLP with the currently available libraries and packages.

The project's sets out to sift through the 36+ million tweets in the dataset in order to successfully detect those tweets that are specifically ***political misinformation***. Once those have been identified, they will be used to construct a ML classifier which can then be used on other, unseen data.

Conceptually, this means we build a funnel to go from:




Which then also becomes the diagram of our workflow:


Finally, the project is also a way for me to accomplish two technical goals:

1. Get familiar with the tools and processes necessary to conduct Arabic NLP
2. Work with **distributed processing** in order to process this larger-than-memory (35GB) dataset.

# Data Wrangling

Besides the standard and universal NLP data wrangling like removing URL's, emoji, hashtags, repeating characters, etc. this project also involved some pre-processing specific to Arabic NLP.

In a blog post on Medium, I provide an in-depth tutorial and explanations as to the particular challenges of working with Arabic text. In short, Arabic has unique characteristics that can lead to extreme data sparsity when processing it in ML models. For this reason, some extra pre-processing steps are needed, which are made possible by the helpful **camel-tools** python package.

The steps taken are:
- Orthographic normalisation
- Dediacritization
- Morphological disambiguation, and
- Lemmatization

# Exploratory Data Analysis

While the bulk of the exploratory data analysis takes place in the Topic Modelling stage, it's worthwhile to mention a number of overall statistics and features of the dataset.

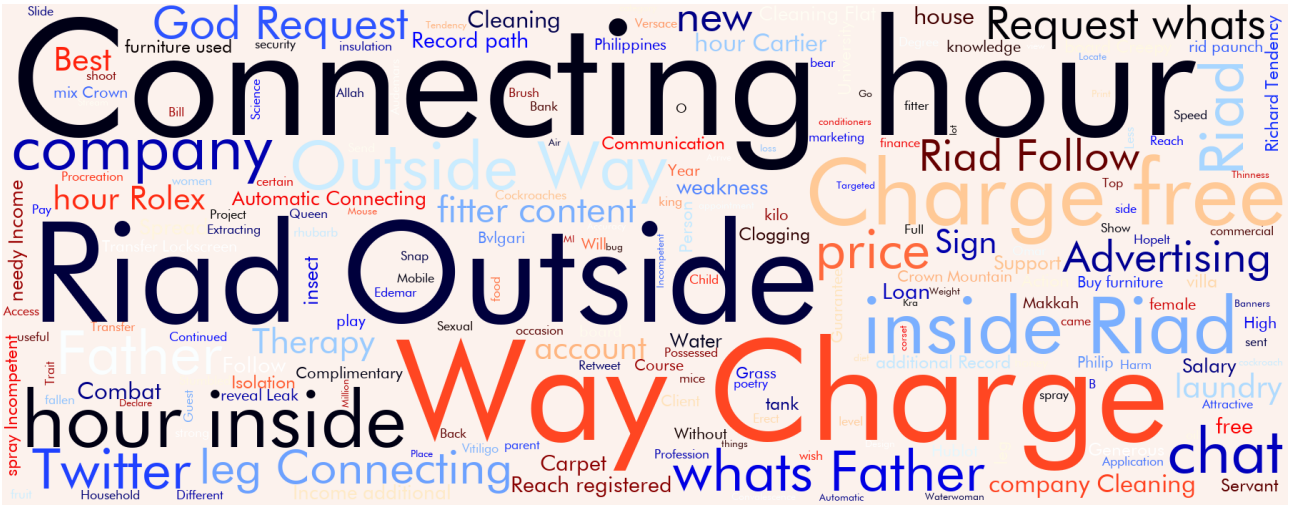
The dataset contains:
- 35.3 million Arabic tweets
- **336,755 unique users** (of whom 4,273 flagged, rest only retweeted)
- Tweets were published between **February 15, 2010** and **January 22, 2020** (most tweets in 2018 and 2019)

Looking at the distribution of tweets, followers, and followings across users, we also notice:
- # Followers per user: ranging from 0 - 1,200,000 -- median: 100
- # Following per user: ranging from 0 - 877,000 -- median: 228
- Top 1% (43) most-followed users produced **almost half of the 35M tweets**
- Tweets per user: ranging from 1 - 1.4 million (**~4.2% of the whole dataset**) -- median: 203

And finally, looking at the unique tweets, we see:
- 6.1M unique tweets
- more than 4M unique tweets show up only once in the dataset
- 78 unique tweets that show up more than 10,000 times
- these **78 unique tweets** together make up **4.5% of the dataset** (more than 1.5M tweets)

This is all indicative of **extreme amplification**: a small number of users and tweets being overrepresented in the dataset.

A closer look at the top100 most recurring tweets also revealed large amounts of non-political content. Below a word cloud of the words included in those top-100 tweets:




The EDA revealed 3 overarching clusters of content:

1. Religious
2. Commercial
3. Political

## Topic Modelling

We employed 2 topic modelling approaches in order to sift out the specifically political content: **Latent Dirichlet Allocation** (LDA) and **Gibbs Sampling Dirichlet Multinomial Mixture** (GSDMM).

While LDA is the most popular topic-modelling approach, it assumes multiple topics per document. GSDMM, on the other hand, is built specifically for short-text clustering and assumes only a single topic per document. Seeing as our documents are max. 140 characters long, it seems reasonable to assume that they contain only a single topic.

Multiple iterations of both modes were done and their performance compared. The GSDMM models clearly did a better job of identifying coherent topics and detecting the political ones. However, the GSDMM iterations were run on a subsample (100k tweets) of the data and when we attempted to pass the GSDMM model the full dataset of 6M unique tweets, the model failed.

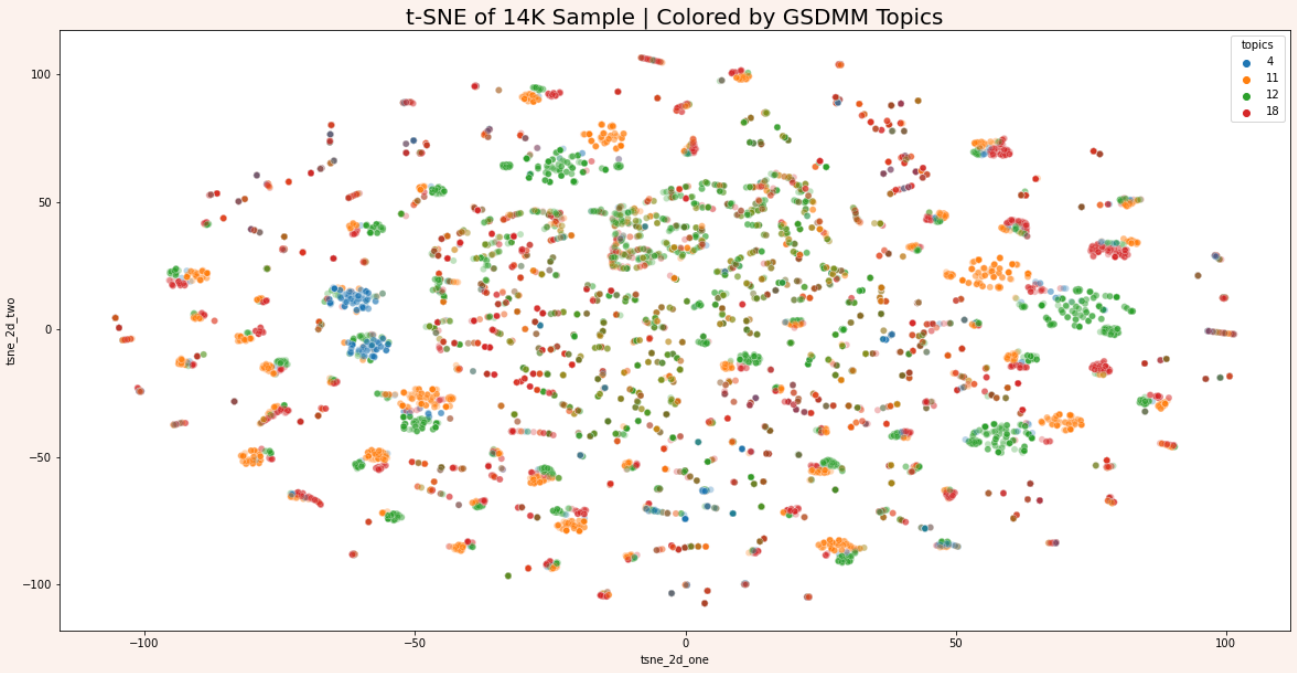
We ended up engineering a work-around in which we took the output from the best LDA model (initiated with 15-topics) — a subset of 350k ‘mostly political tweets — and fed that into our GSDMM model. This was extremely successful and yielded a coherent subset of 285k ‘purely political’ tweets, categorised into 4 sub-topics.

# Clustering

We then set out to further categorise this subset of 285k political tweets into **misinformation** and **neutral** political tweets.

However, here we ran into the technical limitations of the libraries we were working with, and the time limitations of the project. We were not able to run the entire dataset (285k rows * 10k+ features) through a meaningful clustering algorithm.

Instead, we did manage to run a 5% subsample through a t-SNE, revealing large numbers of small clusters, which seemed pretty effectively defined by the GSDMM Topic label, as can be seen in the screenshot below.



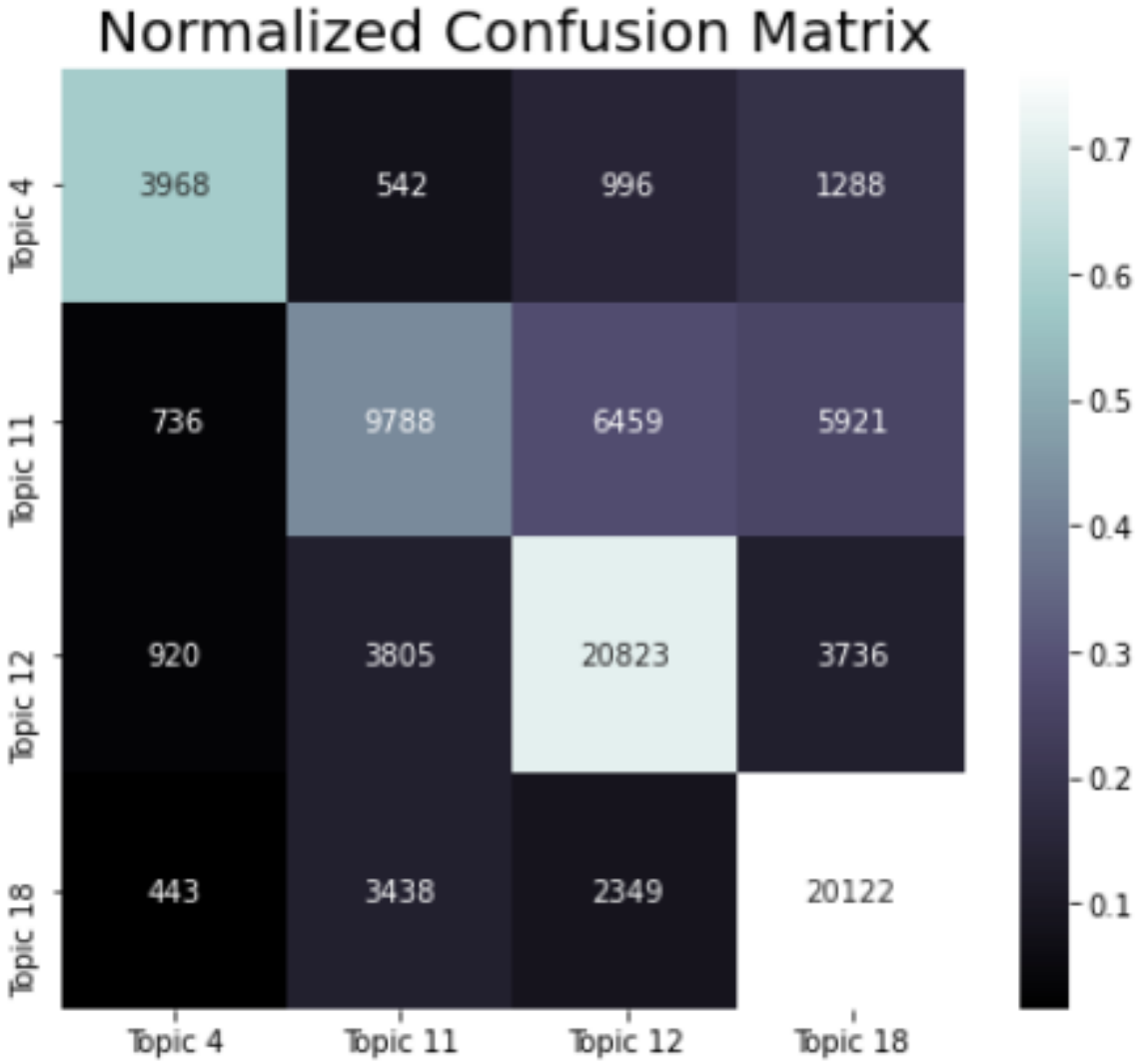

Unfortunately, we have to conclude that we are not able to, at this point in time, to effectively cluster the political tweets into misinformation or neutral. More research and time is needed to figure out the ins-and-outs of running these clustering algorithms in distributed clusters and exploring meaningful ways of teasing the two categories (misinformation or not) out.

# Classification

To complete the technical challenge I set for myself, I did build a Dask implementation of an XGBoost classifier in order to **predict the GSDMM topic label** from all the **non-text features** we had available. I figured this might be an interesting exercise since the topics were shown to be an effective means of clustering the data. If we could predict the topic from the non-text data, that would save some computationally expensive processing steps in our pipeline.

After running an extensive hyper-parameter search, we found the best model could predict the GSDMM topic label from the non-text features with an accuracy, precision, recall, and f1-score of ~63%. While that's not something to write home about, I thought it was still pretty strong predictive power considering this included *nothing* of the original tweet content.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.65 | 0.58 | 0.62 | 6794 |
| 1 | 0.56 | 0.43 | 0.48 | 22904 |
| 2 | 0.68 | 0.71 | 0.70 | 29284 |
| 3 | 0.65 | 0.76 | 0.70 | 26352 |
| accuracy | | | 0.64 | 85334 |
| macro avg | 0.63 | 0.62 | 0.62 | 85334 |
| weighted avg | 0.63 | 0.64 | 0.63 | 85334 |



# Key Takeaways

The project contains (at least) 4 key takeaways in my opinion:

1. Arabic NLP is challenging…but definitely possible, especially with the help of **camel-tools**
2. Topic Modelling on Arabic tweets **works**…and pretty effectively, too!
3. The trade-offs of opting for LDA over GSDMM when working with short-text documents.
4. The world of distributed processing poses a number of extra challenges that require additional time and effort to overcome.

# Future Research

Future research should address:

1. Further exploration of the clustering approach, incl. PCA, distributed t-SNE / Spectral
2. Communication with Python communication regarding possibilities to extend GSDMM to run in distributed mode
3. Extensive network analysis to detect patterns in user interaction and explore potential for the attribution of these information operations to specific nodes in the network.