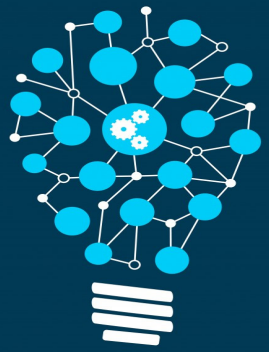
OLD DOMINION UNIVERSITY

MACHINE LEARNING



Model-based Clustering of Short Text Streams

CS722: Machine Learning

Presented by: Roy, Jason, Kamlakant

December 7th, 2018

Introduction

Objective

Propose a single-pass model-based short text stream clustering algorithm.

To deal with:

- The explosive growth of short text in diverse social medias.
  - microblog posts (tweets are an example)
- The concept drift problem and sparsity problem.

Introduction

Challenges

The short text stream clustering problem has the following challenges:

- The sparsity of short text.
- The documents arrive continuously, making it difficult to store all documents and iterate multiple times.
  - This makes static text clustering methods impossible to implement.
- Detecting new clusters and removing outdated clusters automatically
  - This is needed because the topics of the text stream may continuously evolve over time.

Introduction

Schemes

- The problem of clustering short text streams has two schemes: the one pass scheme and the batch scheme

One Pass Scheme

Assumes that the streaming documents come in one by one.
We can process each document only one time (does not allow for iterating).

Batch Scheme

Assumes that the streaming documents come in batches.
We can process the documents in each batch multiple times (allows for iterating, which opens the door for significant performance improvements).

Proposed Model - based Clustering Algorithms

The MStream Algorithm (DPMM Model):

- Has a one-pass clustering process that achieves state of the art results as well as an update clustering process when using batches.
- Improved performance beyond typical state of the art with multiple passes (only possible with batch processing)
  - documents for current batch must be stored

The MStreamF Algorithm:

- Like the MStream algorithm, but incorporates rules for "forgetting" (deleting) outdated data.
- Clusters for each batch have associated meta-data that is used to determine which documents are outdated.

Detailed Algorithm Explanation (MStream Part 1)

//One pass clustering process

for $d = 1$ *to* $|\vec{d}_t|$ **do**

  Compute the probability of document d choosing each of the K existing clusters and a new cluster.

  Sample cluster index z for document d according to the above $K + 1$ probabilities.

  if $z == K + 1$ **then**

    //A new cluster is chosen

    $K = K + 1$

    Initialize m_K, n_K, and n_K^w as zero

  $m_z = m_z + 1$ and $n_z = n_z + N_d$

  for *each word* $w \in d$ **do**

    $n_z^w = n_z^w + N_d^w$

- This part is used alone when simple streaming input is provided
- Compute probabilities of choosing each cluster or a new cluster
- Select a new or existing cluster
- Update the cluster with the information from the new document

Detailed Algorithm Explanation (MStream Part 2)

//Update clustering process
for *iter = 2 to I* **do**
  for *d = 1 to* $|\vec{d}_t|$ **do**
    Record the current cluster of d: $z = z_d$
    $m_z = m_z - 1$ and $n_z = n_z - N_d$
    for *each word* $w \in d$ **do**
      $n_z^w = n_z^w - N_d^w$
    Compute the probability of document d choosing each of the K existing clusters and a new cluster.
    if $iter < I$ **then**
      Sample cluster index z for document d according to the above $K + 1$ probabilities.
    else
      Choose cluster index z for document d with the highest probability.
    if $z == K + 1$ **then**
      //A new cluster is chosen
      $K = K + 1$
      Initialize m_K, n_K, and n_K^w as zero
    $m_z = m_z + 1$ and $n_z = n_z + N_d$
    for *each word* $w \in d$ **do**
      $n_z^w = n_z^w + N_d^w$

- Used with batch scheme.
- Operates similarly to the single pass scheme except that the data from each document in the batch is used to update the probabilities before a final cluster is chosen.
- Documents are allowed to shift clusters from iteration to iteration.
- Eventually, the documents should settle into clusters that best reflect the data in the document, given the data in the other documents in the cluster.
- Reduces the chances of misclassifying a document.

Detailed Algorithm Explanation (MStreamF)

t = 0 // t records the ordinal number of batches.
while *!stream.end()* **do**
  t = t + 1
  $\vec{d}_t$ = stream.next()
  //If the number of stored batches is larger than B_s, we delete the oldest batch from current CF vectors.
  if $t > B_s + 1$ **then**
    $b = t - B_s - 1$
    for *each cluster* $z \in$ *batch* b **do**
      $\vec{n}_z = \vec{n}_z - \vec{n}_{b,z}$
      $m_z = m_z - m_{b,z}$
      $n_z = n_z - n_{b,z}$
    Remove the CF vectors of the oldest batch b.
  //Initialize CF vectors of bath t with current CF vectors.
  $\vec{n}_{t,z} = \vec{n}_z$
  $m_{t,z} = m_z$
  $n_{t,z} = n_z$
  //Clustering documents of batch t with MStream.
  MStream($\vec{d}_t$)
  //Compute CF vectors of batch t
  for *each cluster* $z \in batcht$ **do**
    $\vec{n}_{t,z} = \vec{n}_z - \vec{n}_{t,z}$
    $m_{t,z} = m_z - m_{t,z}$
    $n_{t,z} = n_z - n_{t,z}$

- Keep cluster information for the data of each batch.
- Delete the oldest batch (itself and from cumulative cluster information) if the cache is full.
- Run MStream with the data of current batch and cumulative cluster information.
- Add cluster information for the data of current batch.

Posterior Probability of DPMM model

- Proof in "Jianhua Yin and JianyongWang. 2016. A model-based approach for text clustering with outlier detection. In ICDE. IEEE, 625–636."
  - m_k: $\#documents\ in\ k$
  - N_d^w, N_d: $\#words(w)\ in\ d$
  - n_k^w, n_k: $\#words(w)\ in\ k$
- The probability of document d choosing an existing cluster z:
  - $$p(z = k|D, K - d) \propto \frac{m_k}{|D|-1+\alpha|D|} \cdot \frac{\prod_{w \in d} \prod_{j=1}^{N_d^w} (n_k^w + \beta + j - 1)}{\prod_{i=1}^{N_d} (n_k + V\beta + i - 1)}$$
- The probability of document d choosing a new cluster:
  - $$p(z = |K| + 1|D, K - d) \propto \frac{\alpha|D|}{|D|-1+\alpha|D|} \cdot \frac{\prod_{w \in d} \prod_{j=1}^{N_d^w} (\beta + j - 1)}{\prod_{i=1}^{N_d} (V\beta + i - 1)}$$

The Data Set

Dataset	D	K	V	Avg Len
Tweets & Tweets-T	30,322	269	12,301	7.97
News & News-T	11,109	152	8,110	6.23

Table 1: Statistics of the text datasets (D: Number of documents, K: Number of clusters, V: Vocabulary size, Avg Len: Average length of the documents)

(the "-T" appellation indicates that the data set is sorted and consolidated by Topic)

Preprocessing:

- All letters converted to lowercase, stop words removed (e.g. a, an, the, etc. - words with little semantic weight), stemming performed (words with similar roots collected together).

Evaluation:

- End result of clustering is compared against "ground truth" recorded in the data set by human evaluators.

Results

#topics:
- Tweets: 300
- News: 170

	Tweets	Tweets-T	News	News-T
MStream	**.844** ± .002	.882 ± .004	**.834** ± .004	.850 ± .004
MStreamF	.823 ± .005	**.923** ± .003	.797 ± .003	**.873** ± .003
DTM	.801 ± .002	.802 ± .001	.793 ± .002	.806 ± .002
DCT-L	.697 ± .002	.669 ± .005	.733 ± .002	.744 ± .004
Sumblr	.689 ± .001	.695 ± .003	.575 ± .005	.720 ± .002

Table 2: NMI results of different methods.

α = 0.03, β = 0.03, #iterations = 10, #batch stored = 1

α = 0.01

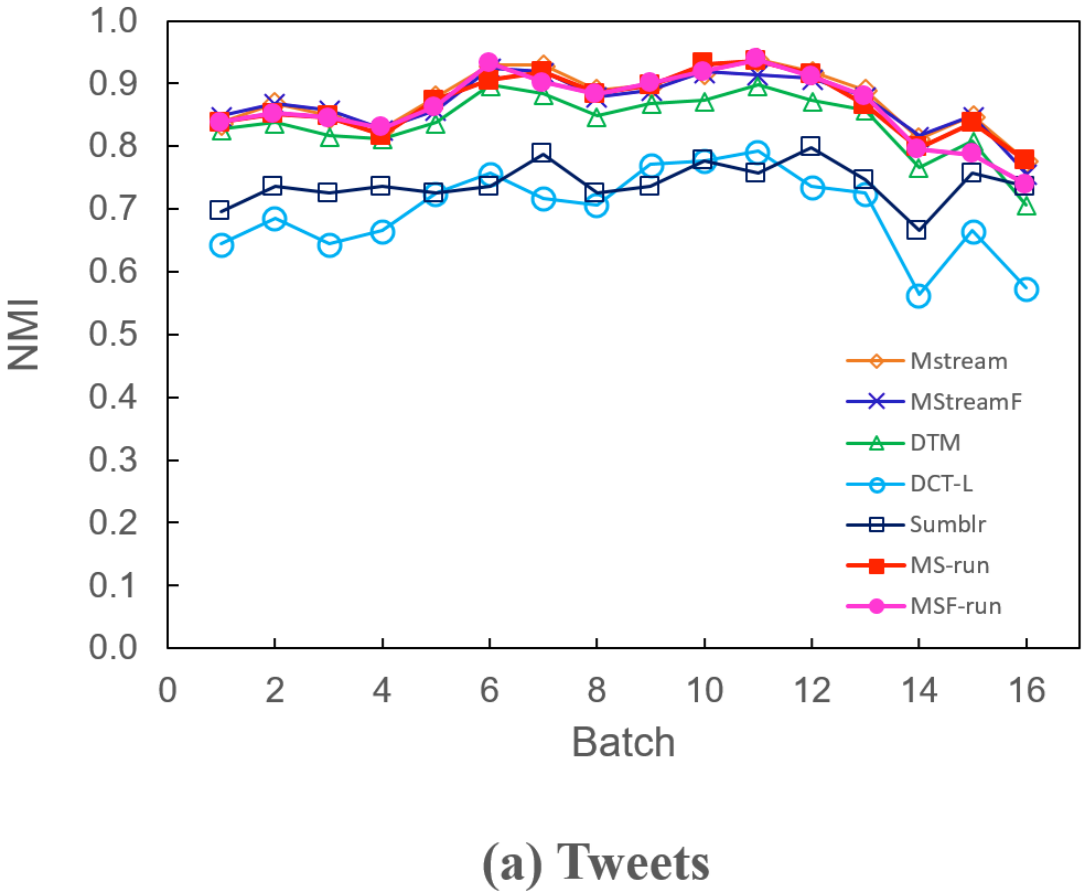
initial α at 1, β at 0.1

initial β to 0.02

- The Normalized Mutual Information (NMI) metric is widely used to evaluate the quality of the clustering results.

$$NMI = \frac{\sum_{c,k} n_{c,k} \log\left(\frac{N \cdot n_{c,k}}{n_c \cdot n_k}\right)}{\sqrt{\left(\sum_c n_c \log \frac{n_c}{N}\right)\left(\sum_k n_k \log \frac{n_k}{N}\right)}}$$

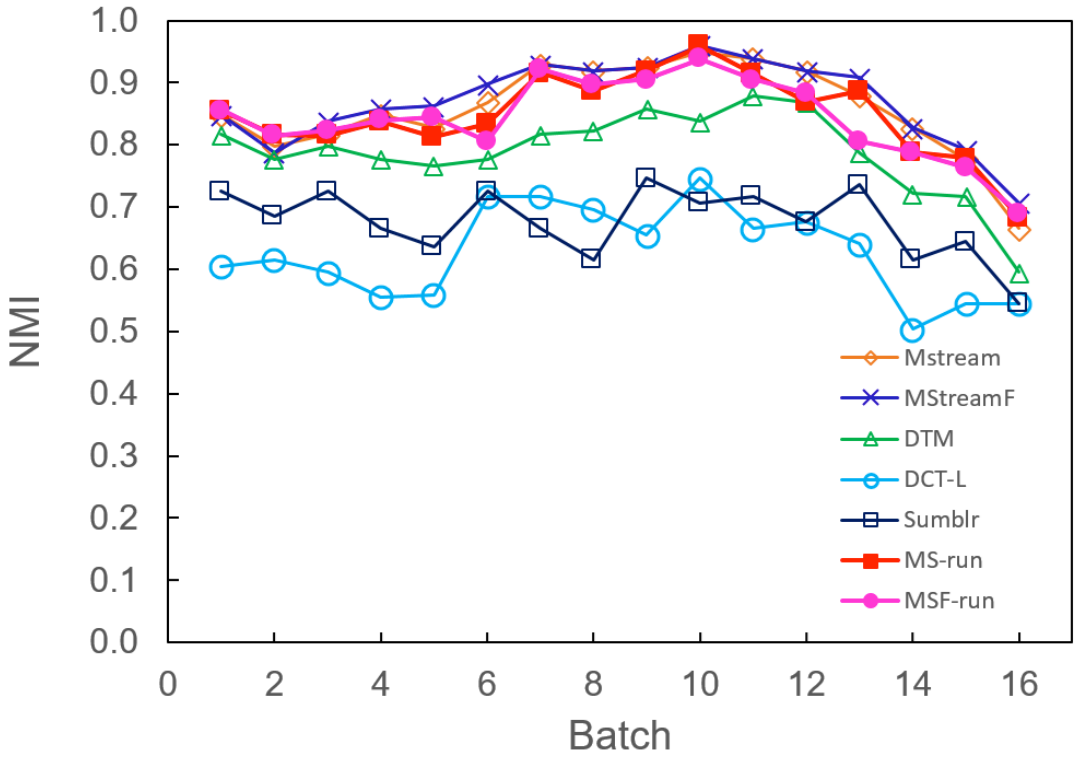
Our Results (1 of 4)



(a) Tweets

- Our results are labeled MS-run and MSF-run, for our versions of MStream and MStreamF.
- Models were coded in Python
- There is some small variation between our results because we used a different random seed.
- Results, however, are highly comparable and in most cases resemble the results in the paper so closely that they obscure the original results on the plot.

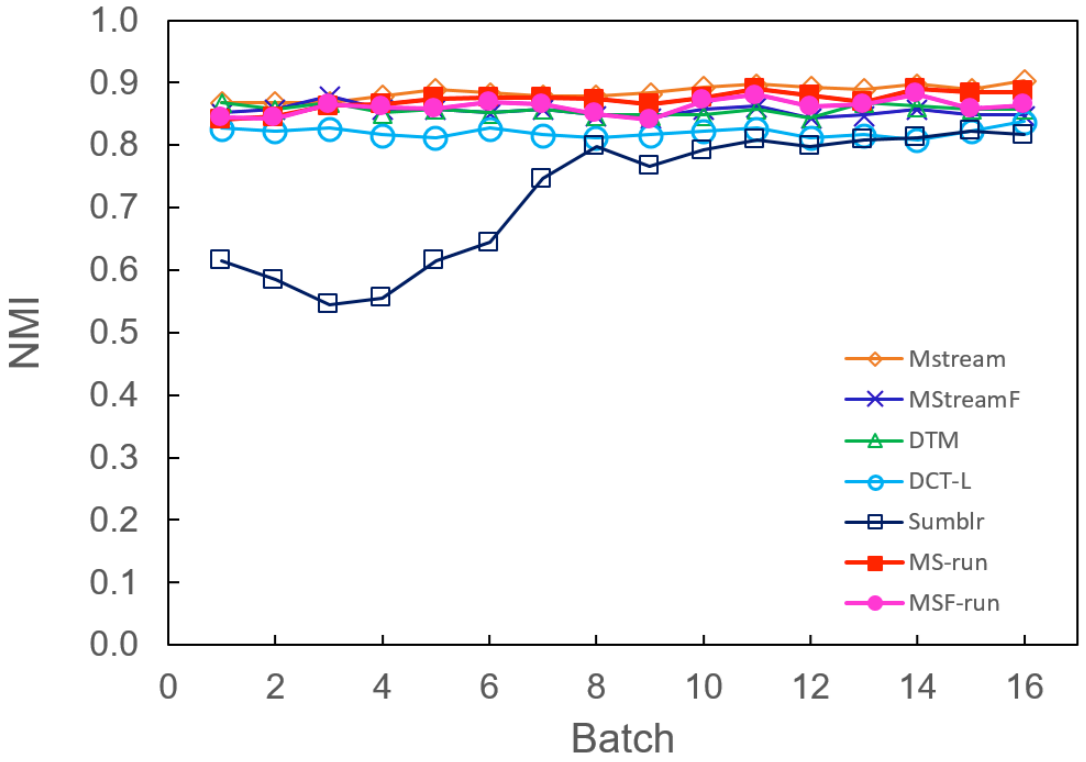
Our Results (2 of 4)



(b) Tweets-T

- Our results are labeled MS-run and MSF-run, for our versions of MStream and MStreamF.
- Models were coded in Python
- There is some small variation between our results because we used a different random seed.
- Results, however, are highly comparable and in most cases resemble the results in the paper so closely that they obscure the original results on the plot.

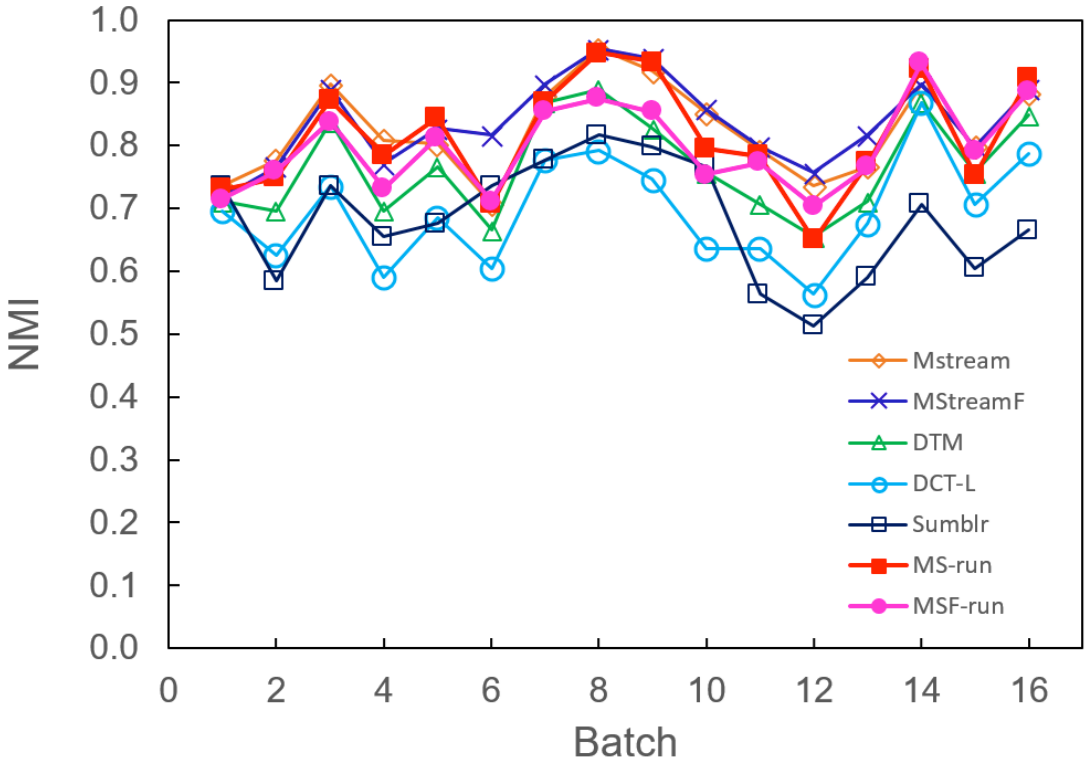
Our Results (3 of 4)



(c) News

- Our results are labeled MS-run and MSF-run, for our versions of MStream and MStreamF.
- Models were coded in Python
- There is some small variation between our results because we used a different random seed.
- Results, however, are highly comparable and in most cases resemble the results in the paper so closely that they obscure the original results on the plot.

Our Results (4 of 4)



(d) News-T

- Our results are labeled MS-run and MSF-run, for our versions of MStream and MStreamF.
- Models were coded in Python
- There is some small variation between our results because we used a different random seed.
- Results, however, are highly comparable and in most cases resemble the results in the paper so closely that they obscure the original results on the plot.

Conclusion

Our results seem to validate those presented in the paper, and MStream/MStreamF show distinct advantages with respect to:

- The ability to handle sparse short text.
- The ability to analyze documents as they arrive continuously, without having to store all documents and iterate multiple times.
- The ability to detect new clusters and remove outdated clusters automatically.

Additionally:

- The MStream/MstreamF algorithms outperform current widely accepted algorithms in terms of both speed and accuracy (as measured by the NMI metric).

Thank you

Backup Slides

Dirichlet process multinomial mixture (DPMM) model

- The Dirichlet distribution is a commonly-used conjugate prior distribution over multinomial distributions
- Dirichlet Processes are a type of Stochastic Process
- Stochastic Process explanation
- Chinese restaurant process

Stochastic Process explanation

- Random Distribution: $P\{X \leq x\} = F(x)$
- Stochastic Process: $P\{X_t \leq x\} = F(x,t)$
- Example:
  - $X_t = 1(awake),\ X_t = 0(asleep)$
  - $P\{X_t = 1\} = F(1,t) = \begin{cases} 90\%,\ if\ 8 < t < 24 \\ 10\%,\ if\ 0 \leq t \leq 8 \end{cases}$
  - $P\{X_t = 0\} = F(0,t) = \begin{cases} 10\%,\ if\ 8 < t < 24 \\ 90\%,\ if\ 0 \leq t \leq 8 \end{cases}$

Chinese restaurant process

- A restaurant with infinite number of tables
- Each table can seat infinite number of customers
- Table k has n_k customers
- The first customer sits at the first table
- Then the n-th customer
  - either chooses an already occupied table k with probability $\frac{n_k}{\alpha+n-1}$
  - or chooses a new table k with probability $\frac{\alpha}{\alpha+n-1}$

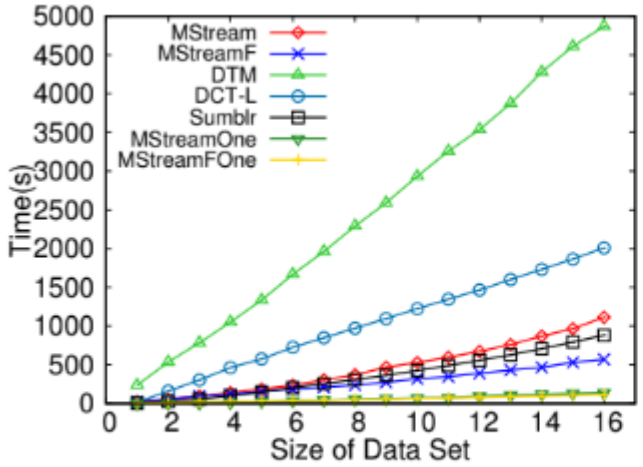
- Only considers the size of the table, ignores the similarity of customers

Naive Bayes assumption

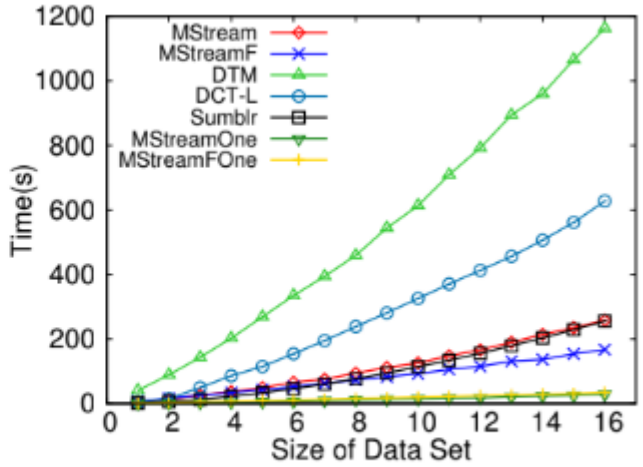
- For any document of any cluster, the probability that it has a certain word is independent of its position and other words it has
  - $p(d|z=k) = \prod_{w \in d} p(w|z=k)$
  - $p(z=k|D, K-d) = \frac{p(D,z=k,K-d)}{p(D,K-d)} \propto \frac{p(D,z=k,K-d)}{p(D-d,K-d)} = \frac{p(D|K)\cdot p(K)}{p(D-d|K-d)\cdot p(K-d)}$
- Any document is a multinomial distribution over words
  - For any position of the document: $\sum_w p(w|d) = 1$
  - Prior distribution for the words: $p(w|z=k) = p(w|z=k,\Phi) = \phi_{k,w},\ \Phi \sim Dir(\vec{\phi_k}, \vec{\beta})$
- Prior distribution for the clusters: $p(z=k) = p(z=k|\Theta) = \theta_k,$

  $\sum_k \theta_k = 1,\ \Theta \sim Dir(\vec{\theta}, \vec{\alpha})$

Speed Results



(a) Speed on Tweets



(b) Speed on News

- The speed of analysis using the MStream and MStreamF family of algorithms showed great improvement over the current state of the art DTM and DCT-L algorithms.
- Clustering results for MStream and MStreamF algorithms met or exceeded the clustering accuracy of the state of the art algorithms.

Additional Results (NMI explanation)

$$NMI = \frac{\sum_{c,k} n_{c,k} \log\left(\frac{N \cdot n_{c,k}}{n_c \cdot n_k}\right)}{\sqrt{\left(\sum_c n_c \log \frac{n_c}{N}\right)\left(\sum_k n_k \log \frac{n_k}{N}\right)}}$$

- The NMI measures the amount of statistical information shared by the random variables representing the cluster assignments and the ground truth groups of the documents.
  - n_c = number of documents in class "c"
  - n_k = number of documents in cluster "k"
  - $n_{c,k}$ = the number of documents in class "c" that are also in cluster "k"
  - N = number of documents in the dataset

Additional Results (Model Summaries)

- DTMs (Dynamic Topic Models) are generative models that can be used to analyze the evolution of (unobserved) topics of a collection of documents over time. This family of models is an extension to Latent Dirichlet Allocation (LDA) that can handle sequential documents.
- DCT-L (Dynamic Clustering Topic model) enables tracking the time-varying distributions of topics over documents and words over topics. Long-term dependency DCT (DCT-L) can capture long-term trends in topics.
- Sumblr proposes an online tweet stream clustering algorithm which is able to efficiently cluster the tweets and maintain compact cluster statistics, with only one pass of the stream.