Dynamic Dirichlet-Multinomial Mixture Models for Analyzing the Time Evolution of Topics in Short Texts

Kevin Christian Wibisono and Yuki Nishimura
Github repo: https://github.com/kwibisono/DDMM_Youtube

1 Introduction

Everyday a vast amount of information is created; and in this dynamically changing world, it is becoming increasingly difficult to pick out important information. To extract and understand information at a more digestible level, researchers have been applying clustering methods to various types of domains. These methods work effectively when the target data is static or accumulates slowly. But in fast-evolving domains such as news and online videos, clustering should be conducted dynamically. In addition, data originating from those domains tend to have limited amount of information, since there is less time for data accumulation. This introduces the need for the clustering methods to be able to deal with short texts, which are inherently sparse.

To deal with the aforementioned issues, we refer to the work of Duan et. al (2018) in which they combine the ideas of Dynamic Topic Models (DTMs), which cluster dynamically, and Dirichlet-Multinomial Mixture Models (DMMs), which cluster short text, thus resulting in a model called the Dynamic Dirichlet-Multinomial Mixture Model (DDMM). We derive and implement a collapsed Gibbs sampler for posterior inference of the model, apply it to the Trending YouTube Video Statistics dataset available on Kaggle, and subsequently conduct quantitative and qualitative analyses of the resulting clusters. We find that a slight modification of the original DDMM algorithm leads to a significant model improvement.

2 Background

2.1 Latent Dirichlet Allocation (LDA) (Blei et al., 2003)

Presumably the most well-known algorithm for topic modeling, LDA assumes that each document consists of a mixture of topics, where each topic has its own distribution of words. The generative process and graphical representation for LDA are shown in Appendix A (Figure 1). For posterior inference of the latent variables, sampling-based methods such as Gibbs sampling or optimization-based methods such as variational inference are typically utilized. The assumption of each document being a mixture of topics holds well in settings where there is abundant text within each document; but in short text settings where there is insufficient text per document, the assumption may not hold well.

2.2 Dynamic Topic Model (DTM) (Blei and Lafferty, 2006)

DTM is a topic model that was developed to analyze the dynamic evolution of topics in large document collections. State space models are used on the parameter space of topic multinomials and topic proportions, where the evolution of states over time reflect the dynamic nature of topics. The generative process and graphical representation for DTM are shown in Appendix A (Figure

2). Since DTM is composed of Latent Dirichlet Allocation (LDA) modules at each time step, it may not perform well in short text settings.

2.3 Dirichlet-Multinomial Mixture Model (DMM) (Yin and Wang, 2014)

DMM is a probabilistic generative model for documents similar to LDA, except that the assumption is that each document consists of only a single topic. The generative process and the graphical representation for DMM are shown in Appendix A (Figure 3). Intuitively, as short texts tend to consist of a single (or a very small number of) topics, DMM is known to perform well in such settings. In addition, DMM is also known to be able to find the number of clusters automatically.

3 Dynamic Dirichlet-Multinomial Mixture Model (DDMM) (Duan and Li, 2018)

A natural solution to dealing with limited and dynamic data is to combine the advantages of DTM and DMM, which leads to the utilization of DDMM. Similar to DTM, DDMM assumes that topic multinomials and cluster proportions evolve over time; and as an extension of DMM, DDMM contains DMM modules within each time slice. The generative process could be described as follows for each time t (where $1 \leq t \leq T =$ number of time steps):

1. Draw cluster proportion $\theta_t \sim \mathrm{Dir}_K(\alpha_t)$
2. For each cluster $k \in \{1, 2, \cdots, K\}$:
   (a) Draw cluster component $\phi_{t,k} \sim \mathrm{Dir}_V(\beta_t)$
3. For each document $d \in \{1, 2, \cdots, D^{(t)}\}$,
   - Draw cluster assignment $z_d \mid \theta_t \sim \mathrm{Cat}(\theta_t)$.
   - Draw words $x_{di} \mid \phi_t, z_d \sim \mathrm{Cat}(\phi_{t,z_d})$.

The graphical representation for DDMM is shown in Figure 4. A key point of DDMM is that the hyperparameters to the exchangeable K and V-Dirichlet α and β are adjusted across time as governed by the following equations:

$$\alpha_k^{(t+1)} = \alpha_k^{(t)} + \lambda \mathrm{p}(m_k^{(t)}); \tag{1}$$

$$\beta_{kv}^{(t+1)} = \beta_{kv}^{(t)} + \mu \mathrm{p}(n_{kv}^{(t)}), \tag{2}$$

where $\mathrm{p}(m_k^{(t)}) = m_k^{(t)}/D^{(t)}$, $\mathrm{p}(n_{kv}^{(t)}) = n_{kv}^{(t)}/n_k^{(t)}$, and λ and μ indicate the amount of influence from the previous time step. Here, D denotes the number of documents, m_k denotes the number of documents belonging to cluster k, n_{kv} denotes the number of occurrences of vocabulary v in all documents belonging to cluster k, and n_k denotes the number of all words in all documents belonging to cluster k. The case where $\lambda = \mu = 0$ can be viewed as having T DMMs trained separately, one for documents in each time step. For posterior inference, we apply the collapsed Gibbs sampler; more details of the derivation (including the full algorithm) are detailed in Appendix C.

We discover that the terms $\mathrm{p}(m_k^{(t)})$ and $\mathrm{p}(n_{kv}^{(t)})$ do not make sense as they result in a severe underestimation of the effect of cluster assignments at the previous time step. In order to mitigate this potential problem, we modify Equations (1) and (2) to be the following: $\alpha_k^{(t+1)} = \alpha_k^{(t)} + \lambda m_k^{(t)}$ and $\beta_{kv}^{(t+1)} = \beta_{kv}^{(t)} + \mu n_{kv}^{(t)}$. For simplicity, we assume $\lambda = \mu = 1$ throughout this paper.

4 Experiment Setting

We apply our modified DDMM (abbreviated to MDDMM) to the Trending YouTube Video Statistics dataset available on Kaggle. The US dataset consists of short texts of up to the top 200 trending videos per day in the US across approximately 8 months. We focus on the fields *title*, *tags*, and *description*, and create 7 datasets that consist of all 7 combinations of those fields (e.g. *title* dataset, *tags_description* dataset, *title_tags_description* dataset). For the preprocessing of each dataset, we remove YouTube specific and NLTK-defined stopwords, URLs, punctuation, and numbers, then convert the words to lower case and lemmatize them. In addition, we filter out words that are not within the top 2500 most frequent words. Ground truth labels of the clusters are given in the *genre* field. We implement the collapsed Gibbs sampling algorithm for MDDMM using Python 3.6, and conduct quantitative and qualitative analysis including other models such as LDA, DMM, DDMM, and DTM. We use our own implementation for DMM, DDMM, and MDDMM, and the `gensim` implementation for LDA and DTM.

5 Results and Discussion

5.1 Quantitative Analysis

To evaluate clusters quantitatively, we focus on Adjusted Mutual Information (AMI), a metric largely used for imbalanced clusters. Simply put, it evaluates how accurate the data is clustered by comparing them to ground truth clusters while considering the imbalance of the clusters, correcting for the effect of agreement due to chance. The paper by Nguyen et al. (2010) provides a rigorous treatment of AMI. Throughout this section, one time step corresponds to one month. Since each of our datasets spans across 8 months, we would have 8 time steps. Also, unless stated otherwise, we set $K = 16$ since the ground truth consists of 16 different clusters.

5.1.1 Comparing LDA and DMM Performance

In order to compare the performance of LDA and DMM in clustering short texts, we run each algorithm 5 times on the *title_tags_description* dataset using default parameters (`gensim`'s default parameters for LDA and $\alpha = \beta = 0.1$ for DMM [as in Yin and Wang, 2014]) and compute the AMI for each run. Each run of DMM consists of at most 5 epochs, with an early stopping criterion when cluster assignments do not change for at least 97% of all documents (i.e. when clusters are "stable"). Table 1 shows that DMM is a better model in terms of AMI, validating the hypothesis that DMM performs better on short texts. Also, it takes about 4 epochs on average for DMM to obtain stable clusters, demonstrating that DMM is fast to converge.

Model name	Mean of AMI	SD of AMI
LDA	0.191	0.011
DMM	**0.218**	0.014

Table 1: Summary of AMI scores for LDA and DMM

5.1.2 Comparing DTM, DDMM, and MDDMM Performance

In order to compare the performance of DTM, DDMM, and MDDMM in clustering short texts dynamically, we run DDMM and MDDMM 5 times, and DTM 2 times (due to slow convergence)

on the *title_tags_description* dataset using default parameters ($\alpha = 1.3$ and $\beta = 0.02$ for DDMM [as in Duan and Li, 2018] and `gensim`'s default parameters for DTM). Similar to the previous experiment, we compute the AMI for each run, which consists of at most 5 epochs with the same early stopping criterion. Table 2 shows that MDDMM achieves a much higher AMI than DDMM, but a lower AMI than DTM. It is highly possible that carefully tuning the hyperparameters might result in a significantly better performance of MDDMM considering the superior performance of DMM in clustering short texts. However, it is worth mentioning that MDDMM is very fast to converge, taking only 2-3 epochs per time step to obtain stable clusters and having an overall running time of around 7 minutes per run (as compared to around 24 hours per run for DTM).

Model name	Mean of AMI	SD of AMI
DTM	**0.268**	0.023
DDMM	0.021	0.002
MDDMM	0.175	0.016

Table 2: Summary of AMI scores for DTM, DDMM and MDDMM

5.1.3 Studying the Convergence Rate of MDDMM on Different Datasets

In order to study the convergence rate of MDDMM on different datasets, we run the algorithm for 5 epochs using default parameters $\alpha = 1.3$ and $\beta = 0.02$, and no early stopping criterion on 3 datasets: *title*, *title_tags* and *title_tags_description*. For each time step, we measure the cluster similarity between the last 2 epochs, which is defined as the proportion of documents having the same cluster assignment in the last 2 epochs. From Figure 5, we observe that DDMM is generally very fast to converge, as evidenced by the clusters produced in the last 2 epochs being mostly similar for each dataset and time step. Also, we find that the similarities tend to increase as the time step increases (especially for time step 3 to 7), which may be explained by the fact that larger time steps correspond to more informative values of α and β. Lastly, we infer that richer datasets (i.e. those with longer text fields, but still considered short text) tend to result in stable clusters more quickly.

5.1.4 Studying MDDMM Performance with Different K Values on Different Datasets

In order to study the performance of MDDMM with different values of K (number of clusters) on different datasets, we run the algorithm 5 times each on each of the 7 datasets with $K \in \{8, 16, 32\}$ using the same experiment settings as in Section 5.1.2. Figure 6 shows that the AMI scores typically increase as K increases. Also, the datasets *tags_description* and *title_tags_description* produce the highest AMI scores, while the dataset *title* produces the lowest AMI scores across different values of K. In general, it seems that richer datasets tend to have higher AMI scores. Table 3 provides a more detailed summary of this result which includes the standard deviation of each AMI value.

K | Dataset	Title	Tags	Description	Title Tags	Title Description	Tags Description	Title Tags Description
8	(0.025, 0.004)	(0.103, 0.017)	(0.110, 0.002)	(0.119, 0.018)	(0.104, 0.009)	(0.153, 0.013)	(0.128, 0.024)
16	(0.031, 0.002)	(0.126, 0.006)	(0.125, 0.011)	(0.113, 0.006)	(0.135, 0.013)	(0.170, 0.012)	(0.175, 0.016)
32	(0.042, 0.004)	(0.139, 0.013)	(0.134, 0.010)	(0.131, 0.002)	(0.130, 0.008)	(0.181, 0.007)	(0.177, 0.006)

Table 3: AMI scores with different K values on different datasets; format is (mean, sd)

5.2 Qualitative Analysis

To evaluate clusters qualitatively, we compare the topics produced by LDA and DMM on the entire dataset, and compare the dynamic evolution of topics per month produced by DTM and MDDMM, on the *title_tags_description* dataset with $K = 16$. We show the results of models with the highest AMI in the previous experiments.

5.2.1 Comparing LDA and DMM Topics

The topics produced by LDA and DMM are very similar, as shown in the Appendix B Qualitative Analysis section. We can argue that not only does the DMM produce a higher AMI, but it also produces interpretable clusters similar to the widely used LDA.

5.2.2 Comparing DTM and MDDMM Dynamic Topics

We show examples of the dynamic evolution of 3 interpreted topics for both DTM and MDDMM in Figures 7, 8, and 9. In general, it can be observed the MDDMM emphasizes words that are more month-specific; in the *Cooking* topic, the word *chocolate* is emphasized in February when Valentine's day happens for MDDMM, whereas *chocolate* appears in many other months for DTM; in the *DIY* topic, *foil ball* is spotted easier in April, and *guava juice* is spotted easier in June for MDDMM; in the *Seasonal* topic, *black friday* and *christmas* are emphasized in their corresponding months of November and December for MDDMM. The emphasis MDDMM makes on month-specific words may be due to MDDMM having fewer parameters than DTM, which leads to parameter updates with larger magnitude within each timestep. We can argue that despite the inferiority in AMI for MDDMM, dynamic topics with more emphasis on each timestep topic can be captured at a lower expense of time.

6 Future Work

Some possible area for future work include: (1) implementing DDMM using variational inference and studying its performance; (2) experimenting with dataset variants for DTM; (3) experimenting with other short text data sets; (4) examining the impact of different hyperparameter settings (e.g. α, β) on model performance; and (5) analyzing model performance using other evaluation metrics (e.g. homogeneity, completeness, adjusted rand index).

References

Banerjee, S., Ramanathan, K., and Gupta, A. (2007). Clustering Short Texts Using Wikipedia. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 787-788.

Blei, D.M., Ng, A.Y., Jordan, M.I. (2003). Latent Dirichlet Allocation. In *Journal of Machine Learning Research 3*, 993-1022.

Blei, D.M., and Lafferty, J.D. (2006). Dynamic Topic Models. In *Proceedings of the 23rd International Conference on Machine Learning*, 113–120.

Duan, R. and Li, C. (2018). An Adaptive Dirichlet Multinomial Mixture Model for Short Text Streaming Clustering. In *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, 49-55.

Nguyen, X. V. , Epps, J., and Bailey, J. (2010). Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research 11*, 2837-2854.

Shou, L., Wang, Z., Chen, K., and Chen, G. (2013). Sumblr: continuous summarization of evolving tweet streams. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 533-542.

Tsur, O., Littman, A., and Rappoport, A. (2013). Efficient Clustering of Short Messages into General Domains. In *7th International AAAI Conference On Weblogs and Social Media*.

Yin, J. (2013). Clustering Microtext Streams for Event Identification. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, 719-725.

Yin, J. and Wang, J. (2014). A Dirichlet Multinomial Mixture Model-based Approach for Short Text Clustering. In *KDD'14: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 233–242.

Zamani, M., Schwartz, H.A., Eichstaedt, J., Guntuku, S.C., Ganesan, A.V., Clouston, S., and Giorgi, S. (2020). Understanding Weekly COVID-19 Concerns through Dynamic Content-Specific LDA Topic Modeling. In *Proceedings of the Fourth Workshop on Natural Language Processing and Computational Social Science*, 193–198.

Appendix A (Generative Processes and Graphical Models)

LDA

Let K denote the number of clusters (topics), V denote the vocabulary size, and D denote the number of documents from here on. The generative process for LDA could be described as follows:

1. For each topic $k \in [1, ..., K]$
   (a) Draw topic $\beta_k \sim \mathrm{Dir}_V(\eta)$
2. For each document i:
   (a) Draw topic proportions $\theta_i \sim \mathrm{Dir}_K(\alpha)$
   (b) For each word j:
      i. Draw topic assignment $z_{ij}|\theta_i \sim \mathrm{Cat}(\theta_i)$
      ii. Draw word $x_{ij}|\{\beta, z_{ij}\} \sim \mathrm{Cat}(\beta_{z_{ij}})$

where η is a hyperparameter for an exchangeable V-Dirichlet, and α is a hyperparameter for an exchangeable K-Dirichlet.

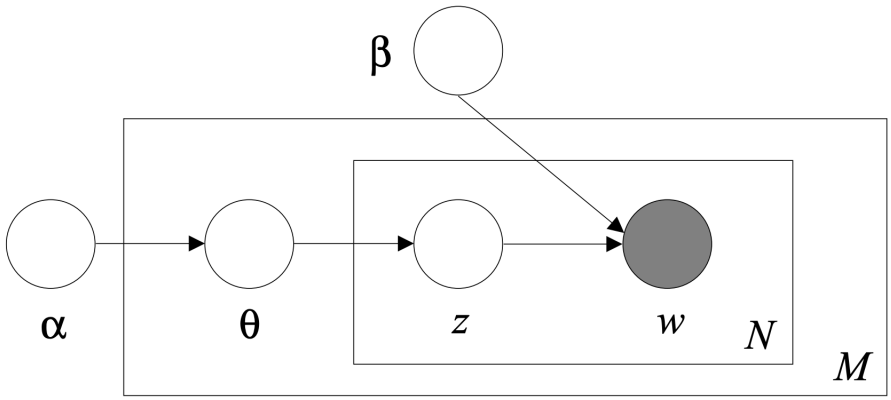

Figure 1: Graphical Representation of Latent Dirichlet Allocation

DTM

The generative process for DTM could be described as follows:

At time t:

(a) Draw topics $\beta_t|\beta_{t-1} \sim N_V(\beta_{t-1}, \sigma^2 I)$

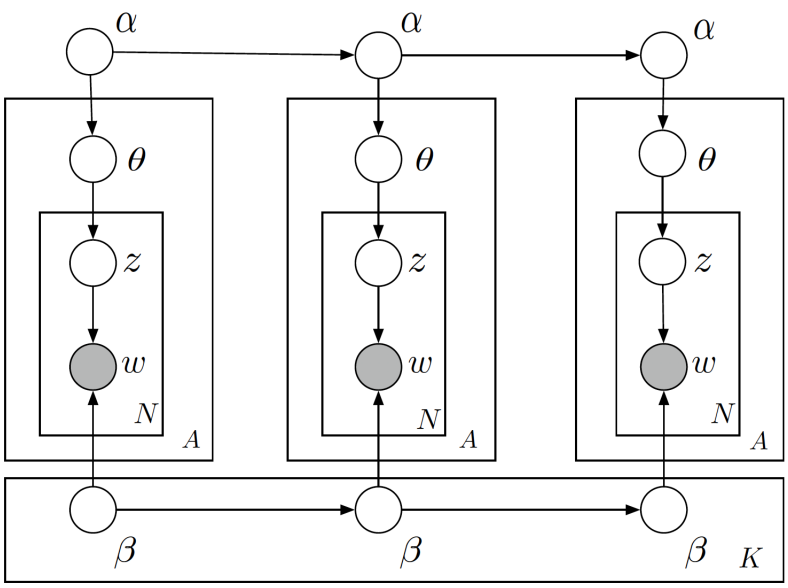

Figure 2: Graphical Representation of Dynamic Topic Model

(b) Draw $\alpha_t|\alpha_{t-1} \sim N_K(\alpha_{t-1}, \delta^2 I)$

(c) For each document $d \in \{1, 2, \cdots, D_t\}$:

i. Draw $\eta_t \sim N_K(\alpha_t, a^2 I)$

ii. For each word j:

A. Draw topic $z \sim \text{Mult}(\pi(\eta_t))$

B. Draw word $w_{t,d,j} \sim \text{Mult}(\pi(\beta_{t,z}))$

where σ, δ, a, are Gaussian variance hyperparameters, and π is a function that maps the Gaussian samples to multinomial parameters. Also, note that the set of documents will differ at each time step.

DMM

The generative process for DMM could be described as follows:

1. Draw cluster proportion $\theta \sim \text{Dir}_K(\alpha)$
2. For each cluster $k \in \{1, 2, \cdots, K\}$:
   (a) Draw cluster component $\phi_k \sim \text{Dir}_V(\beta)$
3. For each document $d \in \{1, 2, \cdots, D\}$,
   - Draw cluster assignment $z_d \mid \theta \sim \text{Cat}(\theta)$.

- Draw words $x_{di} \mid \phi, z_d \sim \text{Cat}(\phi_{z_d})$.

where α is a hyperparameter for an exchangeable K-Dirichlet, and β is a hyperparameter for an exchangeable V-Dirichlet.

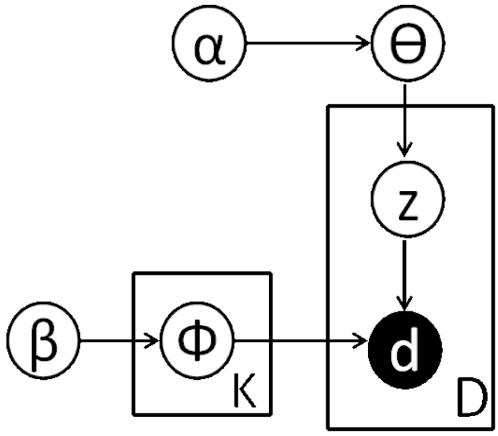

Figure 3: Graphical Representation of Dirichlet Multinomial Mixture Model

DDMM

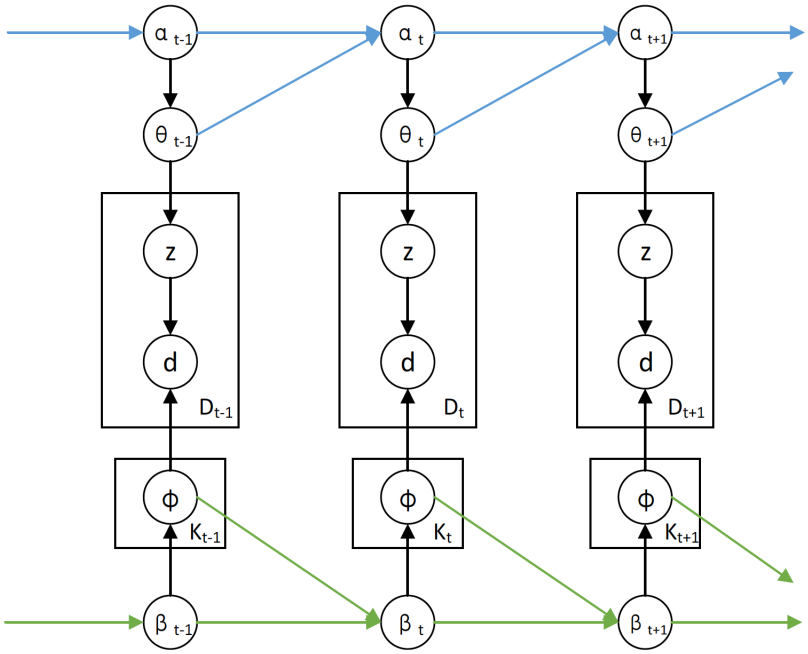

Figure 4: Graphical Representation of Dynamic Dirichlet Multinomial Mixture Model

Appendix B (Experiment Results)

Quantitative Analysis

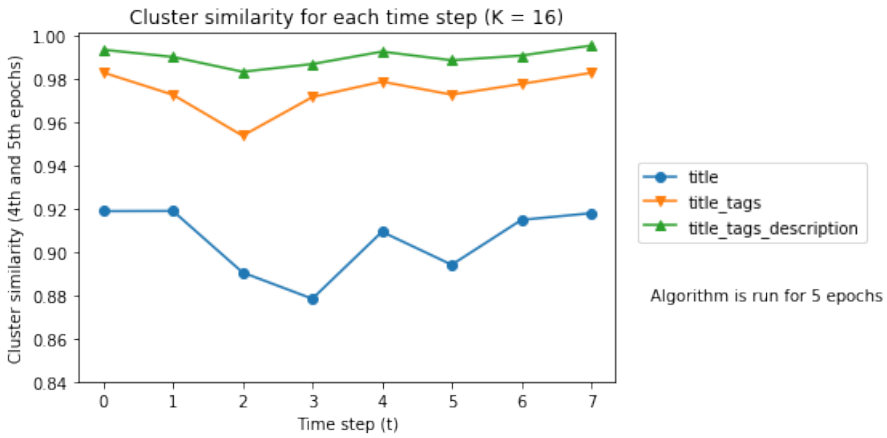

Figure 5: Cluster similarity per time step on different datasets

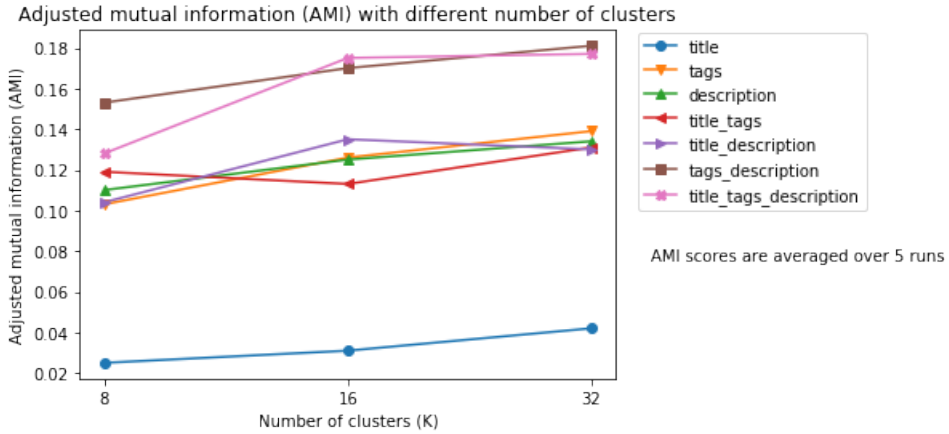

Figure 6: AMI scores with different K values on different datasets

Qualitative Analysis

LDA topics on *title_tags_description* dataset ($K = 16$)

Topic 0: production, buzzfeed, via, inc, warner, chappell, hole, dog, puppy, every

Topic 1: game, news, sport, fox, mendes, shawn, highlight, cup, world, league

Topic 2: cat, black, kid, simon, react, animation, box, brother, fine, panther

Topic 3: night, nba, smith, google, first, wired, take, news, royal, espn

Topic 4: wedding, iphone, netflix, prince, new, diplo, apple, gaming, car, volcano

Topic 5: trailer, movie, film, war, new, star, disney, facebook, deadpool, dan

Topic 6: voice, live, season, episode, snl, american, america, audition, blind, nbc

Topic 7: food, recipe, challenge, test, link, chocolate, cooking, prison, taste, today

Topic 8: challenge, code, use, box, new, infinity, knife, james, cheap, store

Topic 9: room, diy, house, vlog, home, tour, jake, vlogs, day, life

Topic 10: world, wild, science, one, animal, clark, time, get, adventure, radio

Topic 11: show, late, funny, cbs, night, comedy, celebrity, episode, mon, corden

Topic 12: cake, live, jimmy, kimmel, alex, life, abc, bon, dress, brad

Topic 13: show, dude, talent, perfect, tonight, idol, family, jimmy, game, fallon

Topic 14: makeup, getty, beauty, tutorial, fashion, face, hair, product, cosmetic, foundation

Topic 15: know, record, got, love, song, new, need, get, let, high

DMM topics on *title_tags_description* dataset ($K = 16$)

Topic 0: iphone, netflix, new, movie, trailer, smith, apple, galaxy, series, alex

Topic 1: news, tmz, today, time, via, warner, production, inc, chappell, world

Topic 2: cat, cake, simon, new, charlie, game, film, puth, animation, super

Topic 3: news, new, nbc, pop, album, live, camila, facebook, award, record

Topic 4: voice, food, new, street, get, complex, nbc, season, team, sneaker

Topic 5: get, new, know, time, make, science, one, day, people, facebook

Topic 6: trailer, movie, war, star, film, new, black, disney, marvel, facebook

Topic 7: new, link, mythical, record, rhett, song, love, live, gmm, performing

Topic 8: nba, sport, highlight, first, espn, game, wwe, fox, world, take

Topic 9: food, make, recipe, dog, bon, chocolate, challenge, cooking, random, river

Topic 10: dude, perfect, christmas, doctor, new, gane, shot, ever, every, stocking

Topic 11: late, show, night, cbs, funny, nfl, celebrity, seth, snl, comedy

Topic 12: makeup, beauty, code, tutorial, link, produce, use, make, cosmetic, new

Topic 13: react, show, jimmy, tonight, box, fort, fallon, kid, fine, challenge

Topic 14: life, noggin, nail, diy, hair, get, makeover, getty, bbc, new

Topic 15: jimmy, kimmel, live, ellen, idol, show, facebook, american, celebrity, kevin

Topic Evolution Examples on *title_tags_description* dataset ($K = 16$)

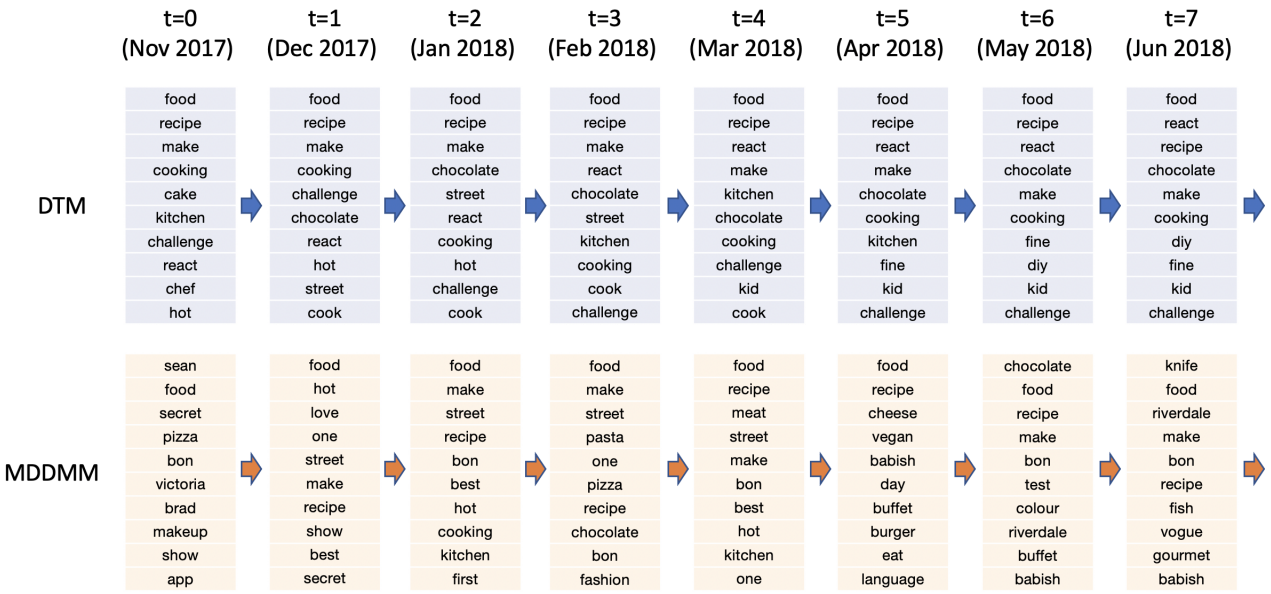

Figure 7: *Cooking* Topic Evolution

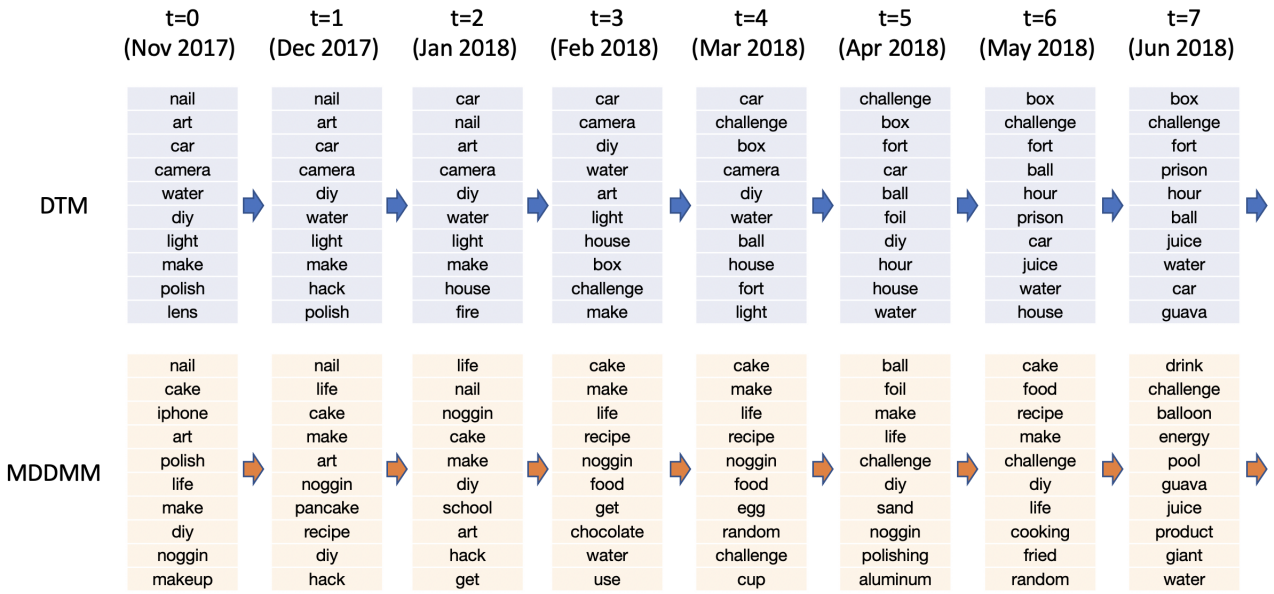

Figure 8: *DIY* Topic Evolution

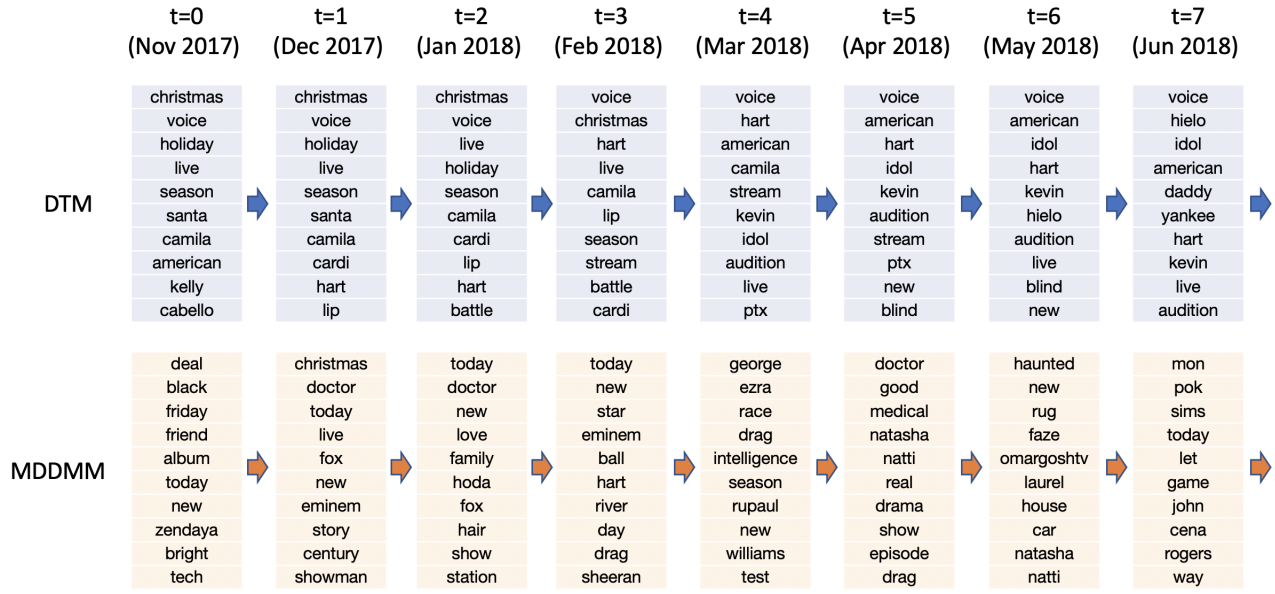

Figure 9: *Seasonal* Topic evolution

Interpreted General Topics per Cluster for DTM and MDDMM

	DTM Topics	MDDMM Topics
Cluster 0	News	Cooking
Cluster 1	Cooking	Smartphones
Cluster 2	Makeup	Music
Cluster 3	Smartphones	News
Cluster 4	Late Night Show/Comedy	Sports/Video Games
Cluster 5	DIY	Movies
Cluster 6	World Adventure	Seasonal
Cluster 7	Sports	Makeup
Cluster 8	Movies	Star Wars/Marvel
Cluster 9	Life Science	Sports
Cluster 10	Animals	Entertainment
Cluster 11	Buzzfeed	Late Night Show/Comedy
Cluster 12	Love	Jimmy Kimmel
Cluster 13	Seasonal	Late Night Show
Cluster 14	Music	DIY
Cluster 15	Star Wars/Celebrity News	Production

Figure 10: Interpreted General Topics for DTM and MDDMM

Appendix C (Collapsed Gibbs Sampler)

DMM

In order to derive a collapse Gibbs sampler algorithm for DMM, it is necessary to compute $\mathrm{p}(z_d = k \mid z_{-d}, x)$, where z_{-d} denotes the cluster labels of all documents except d. Note that we have

$$\mathrm{p}(z_d = k \mid z_{-d}, x) = \frac{\mathrm{p}(x, z \mid \alpha, \beta)}{\mathrm{p}(x, z_{-d} \mid \alpha, \beta)} \propto \frac{\mathrm{p}(x, z \mid \alpha, \beta)}{\mathrm{p}(x_{-d}, z_{-d} \mid \alpha, \beta)}, \tag{3}$$

where x_{-d} denotes all words in all documents except d. The next step is to derive $\mathrm{p}(x, z \mid \alpha, \beta)$, whence $\mathrm{p}(x_{-d}, z_{-d} \mid \alpha, \beta)$ (and the complete conditional) can be inferred very easily. Observe that $\mathrm{p}(x, z \mid \alpha, \beta) = \mathrm{p}(z \mid \alpha)\mathrm{p}(x \mid z, \beta)$.

We can write $\mathrm{p}(z \mid \alpha) = \int \mathrm{p}(z \mid \theta)\mathrm{p}(\theta \mid \alpha)d\theta$. Owing to the fact that $\mathrm{p}(z \mid \theta)$ follows a categorical distribution, we have

$$\mathrm{p}(z \mid \theta) = \prod_{d=1}^{D} \mathrm{p}(z_d \mid \theta) = \prod_{d=1}^{D}\prod_{k=1}^{K} \theta_k^{z_d^{(k)}} = \prod_{k=1}^{K} \theta_k^{\sum_{d=1}^{D} z_d^{(k)}} = \prod_{k=1}^{K} \theta_k^{m_k}, \tag{4}$$

where m_k denotes the number of documents belonging to cluster k. Also, the fact that $\mathrm{p}(\theta \mid \alpha)$ follows a Dirichlet distribution implies that

$$\mathrm{p}(\theta \mid \alpha) = \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \prod_{k=1}^{K} \theta_k^{\alpha - 1}. \tag{5}$$

Therefore, we have

$$\mathrm{p}(z \mid \alpha) = \int \mathrm{p}(z \mid \theta)\mathrm{p}(\theta \mid \alpha)d\theta = \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \int \prod_{k=1}^{K} \theta_k^{\alpha + m_k - 1} d\theta \tag{6}$$

$$= \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \frac{\prod_{k=1}^{K} \Gamma(\alpha + m_k)}{\Gamma(K\alpha + D)} \tag{7}$$

using the fact that $\sum_{k=1}^{K} m_k = D$ and the p.d.f. of a Dirichlet distribution integrates to 1. Similarly, we have $\mathrm{p}(x \mid z, \beta) = \int \mathrm{p}(x \mid z, \phi)\mathrm{p}(\phi \mid \beta)d\phi$. Observe that

$$\mathrm{p}(x \mid z, \phi) = \prod_{k=1}^{K}\prod_{v=1}^{V} \phi_{kv}^{n_{kv}}, \tag{8}$$

where n_{kv} denote the number of occurrences of vocabulary v in all documents belonging to cluster k. Also,

$$\mathrm{p}(\phi \mid \beta) = \prod_{k=1}^{K} \left(\frac{\Gamma(V\beta)}{\Gamma(\beta)^V} \prod_{v=1}^{V} \phi_{kv}^{\beta - 1} \right). \tag{9}$$

Using the same technique as above, we easily obtain

$$\mathrm{p}(x \mid z, \beta) = \prod_{k=1}^{K} \left(\frac{\Gamma(V\beta)}{\Gamma(\beta)^V} \frac{\prod_{v=1}^{V} \Gamma(\beta + n_{kv})}{\Gamma(V\beta + n_k)} \right), \tag{10}$$

where $n_k = \sum_{v=1}^{V} n_{kv}$ denotes the number of words in all documents belonging to cluster k. From here, we have

$$\mathrm{p}(x, z \mid \alpha, \beta) = \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \frac{\prod_{k=1}^{K} \Gamma(\alpha + m_k)}{\Gamma(K\alpha + D)} \prod_{k=1}^{K} \left(\frac{\Gamma(V\beta)}{\Gamma(\beta)^V} \frac{\prod_{v=1}^{V} \Gamma(\beta + n_{kv})}{\Gamma(V\beta + n_k)} \right). \tag{11}$$

Similarly,

$$\mathrm{p}(x_{-d}, z_{-d} \mid \alpha, \beta) = \frac{\Gamma(K\alpha)}{\Gamma(\alpha)^K} \frac{\prod_{k=1}^{K} \Gamma(\alpha + m_{k,-d})}{\Gamma(K\alpha + D - 1)} \prod_{k=1}^{K} \left(\frac{\Gamma(V\beta)}{\Gamma(\beta)^V} \frac{\prod_{v=1}^{V} \Gamma(\beta + n_{kv,-d})}{\Gamma(V\beta + n_{k,-d})} \right). \tag{12}$$

It is easy to see that if document d currently belongs to cluster k, we have $m_{k,-d} = m_k - 1$, $n_{k,-d} = n_k - N_d$ (where N_d denotes the number of words in document d), and $n_{kv,-d} = n_{kv} - N_{dv}$ (where N_{dv} denotes the number of occurrences of vocabulary v in document d). Otherwise, we have $m_{k,-d} = m_k$, $n_{k,-d} = n_k$ and $n_{kv,-d} = n_{kv}$. Combined with the fact that $\frac{\Gamma(x+m)}{\Gamma(x)} = \prod_{i=1}^{m}(x+i-1)$ for any positive integer m, we obtain

$$\mathrm{p}(z_d = k \mid z_{-d}, x) \propto \frac{\mathrm{p}(x, z \mid \alpha, \beta)}{\mathrm{p}(x_{-d}, z_{-d} \mid \alpha, \beta)} \tag{13}$$

$$\propto \left(\frac{\alpha + m_{k,-d}}{K\alpha + D - 1} \right) \frac{\prod_{v=1}^{V} \prod_{j=1}^{N_{dv}} (\beta + n_{kv,-d} + j - 1)}{\prod_{i=1}^{N_d} (V\beta + n_{k,-d} + i - 1)}. \tag{14}$$

From here, the collapsed Gibbs sampler algorithm can be written as follows:

1. Sample a cluster for each document. Compute initial values of m_k, n_k and n_{kv} for each cluster k and vocabulary v.
2. While the Markov chain has not converged, do the following for each document d:
   - "Knock out" document d from its current cluster (let it be z). Modify m_z, n_z and n_{zv} for each vocabulary v accordingly.
   - Sample a cluster for document d according to Equation 15 (let it be q). Modify m_q, n_q and n_{qv} for each vocabulary v accordingly.

DDMM

The dynamic DMM proposed by Duan and Li (2018) is largely similar to the DMM described above, except that the hyperparameters α and β vary across time as governed by the following equations:

$$\alpha_k^{(t+1)} = \alpha_k^{(t)} + \lambda \mathrm{p}(m_k^{(t)}); \tag{15}$$

$$\beta_{kv}^{(t+1)} = \beta_{kv}^{(t)} + \mu \mathrm{p}(n_{kv}^{(t)}), \tag{16}$$

where $\mathrm{p}(m_k^{(t)}) = m_k^{(t)}/D^{(t)}$ and $\mathrm{p}(n_{kv}^{(t)}) = n_{kv}^{(t)}/n_k^{(t)}$. Here, λ and μ indicate the amount of influence from the previous time step. The case where $\lambda = \mu = 0$ can be viewed as having T DMMs trained separately, one for documents in each time step. At the initial time step, $\alpha^{(1)}$ and $\beta_k^{(1)}$ are assumed to be parameters to exchangeable K and V-Dirichlet, respectively. At later time steps, this exchangeability assumption may not hold. Hence, the update rule on Equation 15 becomes

$$\mathrm{p}(z_d = k \mid z_{-d}, x) \propto \left(\frac{\alpha_k + m_{k,-d}}{\left(\sum_{k=1}^{K} \alpha_k\right) + D - 1} \right) \frac{\prod_{v=1}^{V} \prod_{j=1}^{N_{dv}} (\beta_{kv} + n_{kv,-d} + j - 1)}{\prod_{i=1}^{N_d} \left(\left(\sum_{v=1}^{V} \beta_{kv}\right) + n_{k,-d} + i - 1\right)}. \tag{17}$$

The same Gibbs sampler algorithm is used to infer the topic distribution at time t based on that at time $t-1$, for each $1 \leq t \leq T$ (except for $t = 1$, α and β need to be initialized).