CHAINING DATA - A NOVEL PARADIGM IN ARTIFICIAL INTELLIGENCE EXEMPLIFIED WITH NMF BASED CLUSTERING

Anonymous authors
Paper under double-blind review

ABSTRACT

In the era of artificial intelligence, there is an acceleration of high-quality inference from the fusion of data. We have overcome the linking challenge associated with higher-order features. We have fundamentally linked together databases tables for clustering algorithms and expect this paradigm and those related to it to produce many new insights. We propose linked view clustering that extends multi-view clustering by adding complementary and consensus information across linked views of each data point. We focus on non-negative matrix factorization combined with the fusion of linking data corresponding to extracting knowledge from the multiple tables of a relational database. It is commonplace to identify hashtag communities on social media by word usage. However, there exist troves of data not included but could be. We can incorporate locations by hashtag to improve community detection, this is multiNMF or multi-view clustering, but we extend this method beyond the first link. To our knowledge, a general artificial intelligence method to incorporate any table that can be chained backward has not been done before. We call this linked view NMF or chained view clustering and give the algorithms to perform multiplicative updates and the general solution that can be solved using automatic differentiation such as JAX. We demonstrate how the equations can be interpreted on synthetic data and how information flows through the links. As a proof of concept on real data, we incorporate word vectors using the method on an authorship clustering dataset.

1 INTRODUCTION

Most clustering algorithms use either a table or at most two tables to analyze data. However, for those algorithms to be successful, large quantities of relevant data must be incorporated. We will begin with an example, simultaneous community detection and topic modeling using document word clustering on social media data with NMF. Word vectors can be incorporated by adding a word vector table. However, large amounts of data are available that can chain several tables together and are linked, for instance: hashtag word, word location, location author, author topics, as seen in Figure 1. rom this information, we may identify communities that are more specific than simply the words present in the social media posts. At the heart of artificial intelligence is extracting previously unknown and potentially useful information from all available data. However, the problem is that this extent of data fusion using higher-order features has not been done for NMF in prior works.

Although we focus on NMF using multiplicative updates, MU, Seung & Lee (2001) it is entirely possible to define a joint loss function, J, as seen in the three subsections below using automatic differentiation such as JAX, torch, or TensorFlow. The reason to use MU over convex optimizers, alternating nonnegative least squares, and automatic differentiation is the reduced computational load and higher empirically observed quality of the results. However, we have implemented a MU algorithm that uses automatic differentiation that is faster and provides comparable results to MU. To appreciate the challenges that arise in linked view NMF, we must consider that for multi-view NMF, we have a much easier to solve MU and can easily incorporate more significant numbers of views than we could incorporate many linked relations. A multiNMF is where the primary key has features. A multiNMF is where the primary key has features. The difference between multi-view clustering and clustering is that the first treat features sets of similar types, and the latter treats all

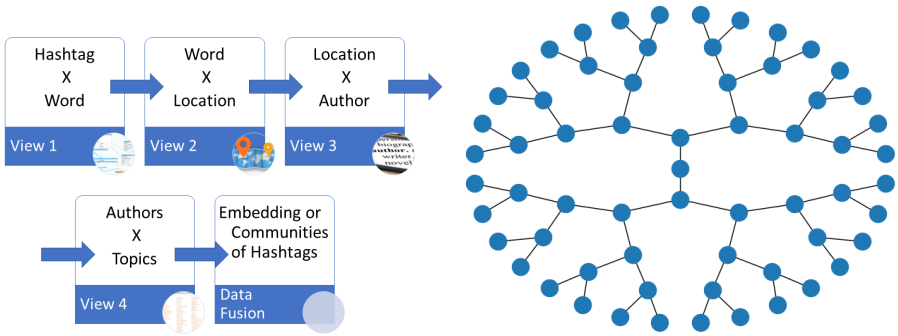

Figure 1: Chaining together views to gain a whole view of multiple interrelated datasets with the only requirement that features are keyed by other features. While a linear chain is shown in the left of the Figure, it can actually be a radial branching chain as seen in the right of the Figure. It is shown symmetric, but there is no requirement for symmetry.

features equally. However, linking relations cannot be captured in traditional clustering directly as they can in the following generalization, which is this work's main contribution.

2 Types of NMF

Here we present first the NMF problem and its MU rule. Next we show the Multiview NMF that factors multiple matrices related by the primary datatype. In this type we give the MU rule and its generalization to any number of primary datatype interrelated matrices. Finally, we illustrate chained view NMF that allows linking of matrices in the same way as a relational database, i.e. feature matrices of features. In linked view we explored the analytic solution but due to computational hurdles we additionally defined an automatic differentiation to ascertain the positive and negative terms of the joint loss function to perform the MU.

2.1 Vanilla NMF

An example of Vanilla NMF is a matrix of hashtags X tf-idf of words and yielding two matrices, W and H. W is the hashtag communities or embeddings and H is the community word usage.

$$Loss = \|\mathbf{V} - \mathbf{WH}\|_F^2 \tag{1}$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\mathbf{VH}^T}{\mathbf{WHH}^T} \tag{2}$$

$$\mathbf{H} \leftarrow \mathbf{H} \odot \frac{\mathbf{W}^T\mathbf{V}}{\mathbf{W}^T\mathbf{WH}} \tag{3}$$

and for KL Divergence from Burred (2014):

$$Generalized\ D_{KL}(P||Q) = \sum_i \left[P(i) \log \frac{P(i)}{Q(i)} - P(i) + Q(i) \right] \tag{4}$$

$$Loss = D_{KL}(\boldsymbol{V}||\boldsymbol{WH}) = \sum_{i,j} \left[V_{i,j} \log \frac{V_{i,j}}{WH|_{i,j}} - V_{i,j} + WH|_{i,j} \right] \tag{5}$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\frac{\mathbf{V}}{\mathbf{WH}}\mathbf{H}^T}{\sum_i \boldsymbol{H}_{i,:}} \tag{6}$$

$$\mathbf{H} \leftarrow \mathbf{H} \odot \frac{\boldsymbol{W}^T \frac{\boldsymbol{V}}{\boldsymbol{WH}}}{\sum_j \boldsymbol{W}_{:,j}} \tag{7}$$

2.2 MULTINMF

For the case of 2 matrices such as hashtag X word and hashtag X location representing V_A and V_B:

$$Loss = \|\mathbf{V_A} - \mathbf{WH_A}\|_F^2 + \|\mathbf{V_B} - \mathbf{WH_B}\|_F^2 \tag{8}$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\mathbf{V}_A\mathbf{H}_A^T + \mathbf{V}_B\mathbf{H}_B^T}{\mathbf{WH}_A\mathbf{H}_A^T + \mathbf{WH}_B\mathbf{H}_B^T} \tag{9}$$

$$\mathbf{H}_A \leftarrow \mathbf{H}_A \odot \frac{\mathbf{W}^T\mathbf{V}_A}{\mathbf{W}^T\mathbf{WH}_A} \tag{10}$$

$$\mathbf{H}_B \leftarrow \mathbf{H}_B \odot \frac{\mathbf{W}^T\mathbf{V}_B}{\mathbf{W}^T\mathbf{WH}_B} \tag{11}$$

for the general case we have a matrices of type X giving us the following equation:

$$Loss = \|\mathbf{V_{X_1}} - \mathbf{WH_{X_1}}\|_F^2 + \|\mathbf{V_{X_2}} - \mathbf{WH_{X_2}}\|_F^2 + ... + \|\mathbf{V_{X_n}} - \mathbf{WH_{X_n}}\|_F^2 \tag{12}$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\mathbf{V}_{X_1}\mathbf{H}_{X_1}^T + \mathbf{V}_{X_2}\mathbf{H}_{X_2}^T + ... + \mathbf{V}_{X_n}\mathbf{H}_{X_n}^T}{\mathbf{WH}_{X_1}\mathbf{H}_{X_1}^T + \mathbf{WH}_{X_2}\mathbf{H}_{X_2}^T + ... + \mathbf{WH}_{X_n}\mathbf{H}_{X_n}^T} \tag{13}$$

$$\mathbf{H}_{X_1} \leftarrow \mathbf{H}_{X_1} \odot \frac{\mathbf{W}^T\mathbf{V}_{X_1}}{\mathbf{W}^T\mathbf{WH}_{X_1}} \tag{14}$$

$$\mathbf{H}_{X_2} \leftarrow \mathbf{H}_{X_2} \odot \frac{\mathbf{W}^T\mathbf{V}_{X_2}}{\mathbf{W}^T\mathbf{WH}_{X_2}} \tag{15}$$

$$... \tag{16}$$

$$\mathbf{H}_{X_n} \leftarrow \mathbf{H}_{X_n} \odot \frac{\mathbf{W}^T\mathbf{V}_{X_n}}{\mathbf{W}^T\mathbf{WH}_{X_n}} \tag{17}$$

2.3 OUR PROPOSED WORK - CHAINED VIEW NMF

What if the table that you would like to incorporate does not have the same type of rows as $\mathbf{V}$ For instance if $\mathbf{V}$ is hashtag×tfIdf of words, then you must use a hashtag table such as hashtag×location. In chained view the requirement is relaxed such that any table can be used that has the same column type as $\mathbf{V}$ for instance $\mathbf{V}_B = word \times wordVector$. What is not immediately apparent is that by relaxing this requirement then any number of possible tables can be joined together for instance: hashtag×tfIdf to word×location to location×author and finally author×topics. Because this follows the principle of foreign keys in a relational database, it is possible to directly use this algorithm to take a whole database and gives clusters based on everything in it. This is in effect a feature of features solution. Using the example from before in the text we formulate the problem as such:

$$\begin{aligned} Loss = &\|\mathbf{V}_A - \mathbf{WH}_A\|_F^2 + \\ &\|\mathbf{V}_B - \mathbf{V}_A^T\mathbf{WH}_B\|_F^2 + \\ &\|\mathbf{V}_C - \mathbf{V}_B^T\mathbf{V}_A^T\mathbf{WH}_C\|_F^2 + \\ &\|\mathbf{V}_D - \mathbf{V}_C^T\mathbf{V}_B^T\mathbf{V}_A^T\mathbf{WH}_D\|_F^2 + \\ &... \end{aligned} \tag{18}$$

Where W is the hashtag communities of the database, $\mathbf{V_A}$ through$\mathbf{V_D}$ are hashtag$X$tfIDF, word$X$location, location$X$author and finally author$X$topics respectively. Interestingly this can also be viewed as an embedding of the database for hashtags. It may be important to decrease the loss

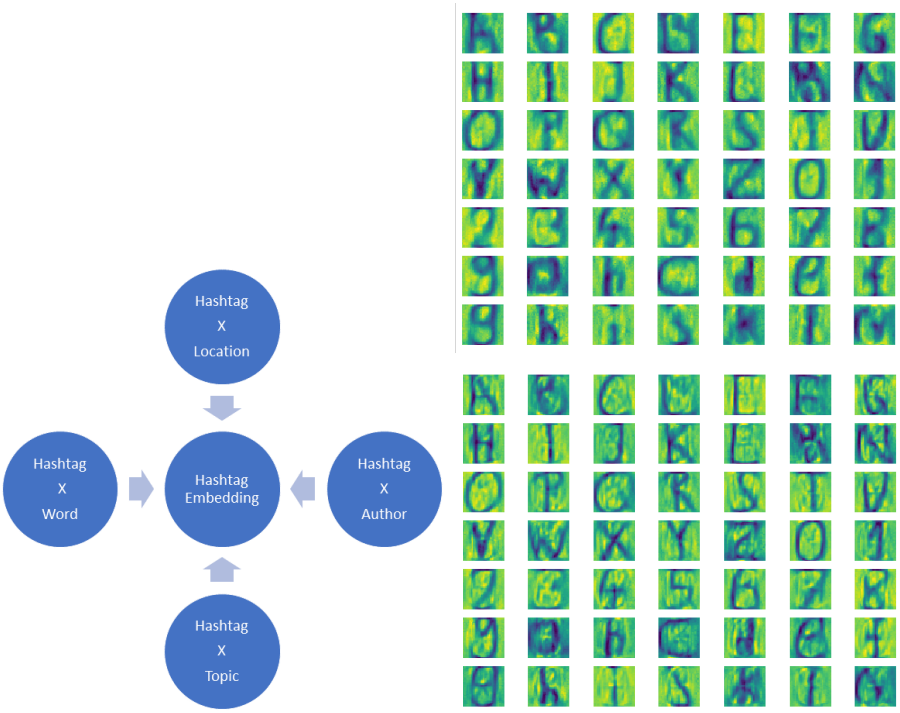

Figure 2: Multi NMF or multiple view clustering. Multiple datasets with the requirement that features are keyed by one embedding or community type, in this case it is the hashtag community or embedding. (Left) Reconstruction of handwritten alphanumerics using multiresolution multiNMF (Right Top) and reconstruction of vanilla NMF on raw data. Instead of LBP counts for each datapoint convolutions are performed for the 16 filters (second resolution NMF H's) are extracted from 16,384 random 8x8 patches. NMF Vanilla on raw data gives NMI and ARI of 0.24 and 0.03. MultiNMF on raw data as well as convolution data gives NMI and ARI 0.35 and 0.10. This is the HASWYv2 dataset. While multiNMF is fuzzier it does cluster better, perhaps in a way filling in the missing parts of letters that it believes it is. We found a similar NMI and ARI for vanilla NMF to multiNMF on the multiresolution data, however it is not possible to reconstruct characters because there is no H corresponding to the nonconvoluted characters. There are 61 possible characters in the dataset.

assigned to increasingly complex chains so they do not contribute to W the same as the most direct chains. Alternatively one may solve for the last term and by doing so incorporate all of the data matrices. Solving for the last term does have one drawback, it disproportionately weighs all data matrices equal, this may not be the intent.

Here is the MU rule for a single hop of the linked view NMF that while equivalent to a transpose of a multiview, this gives the basis for multiple hops.

Algorithm 1 Function: Multiplicative Update Auto Differentiation

Input: Joint Loss Function
Output: Matrix to Multiply by W, H, H2, H3, ..., Hn during update
Initialisation : Define Positive and Negative Terms of the Joint Loss Function
grad : Providing the Gradients through Auto Differentiation i.e. JAX for Specified Variable
Gradients = Absolute Value of All Terms after Application of *grad*
Numerator = Sum of All Negative Terms' Gradients
Denominator = Sum of All Positive Term's Gradients
Denominator += Epsilon of Machine
return Numerator / Denominator

$$Loss = \|\mathbf{V}_A - \mathbf{W}\mathbf{H}_A\|_F^2 + \|\mathbf{V}_B - \mathbf{V}_A^T\mathbf{W}\mathbf{H}_B\|_F^2 \tag{19}$$

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\mathbf{V}_A\mathbf{V}_B\mathbf{H}_B^T + \mathbf{V}_A\mathbf{H}_A^T}{\mathbf{V}_A\mathbf{V}_A^T\mathbf{W}\mathbf{H}_B\mathbf{H}_B^T + \mathbf{W}\mathbf{H}_A\mathbf{H}_A^T} \tag{20}$$

$$\mathbf{H}_A \leftarrow \mathbf{H}_A \odot \frac{\mathbf{W}^T\mathbf{V}_A}{\mathbf{W}^T\mathbf{W}\mathbf{H}_A} \tag{21}$$

$$\mathbf{H}_B \leftarrow \mathbf{H}_B \odot \frac{\mathbf{W}^T\mathbf{V}_A\mathbf{V}_B}{\mathbf{W}^T\mathbf{V}_A\mathbf{V}_A^T\mathbf{W}\mathbf{H}_B} \tag{22}$$

The following is the KL divergence chained linkage MUR. The ones matrix in the shape of V_B is found in Equation 23. The ones matrix in Equations 24 and 25 is the shape necessary to allow element-wise division and products.

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{\boldsymbol{V}_A \frac{\boldsymbol{V}_B}{\boldsymbol{V}_A^T\boldsymbol{W}\boldsymbol{H}_B}\boldsymbol{H}_B^T + \frac{\mathbf{V_A}}{\boldsymbol{W}\boldsymbol{H}_A}\boldsymbol{H}_A^T}{\mathbf{1}\boldsymbol{H}_A^T + \boldsymbol{V}_A\mathbf{1}\boldsymbol{H}_B^T} \tag{23}$$

$$\boldsymbol{H}_A \leftarrow \boldsymbol{H}_A \odot \frac{\boldsymbol{W}^T\frac{\boldsymbol{V}_A}{\boldsymbol{W}\boldsymbol{H}}}{\boldsymbol{W}^T\mathbf{1}} \tag{24}$$

$$\boldsymbol{H}_B \leftarrow \boldsymbol{H}_B \odot \frac{\boldsymbol{W}^T\boldsymbol{V}_A\frac{\boldsymbol{V}_B}{\boldsymbol{V}_A^T\boldsymbol{W}\boldsymbol{H}_B}}{\boldsymbol{W}^T\boldsymbol{V}_A\mathbf{1}} \tag{25}$$

The general case for Frobenius Loss is (up to the third linkage or V_D):

$$\begin{aligned} Loss = &\|\mathbf{V}_A - \mathbf{W}\mathbf{H}_A\|_F^2 + \\ &\|\mathbf{V}_B - \mathbf{V}_A^T\mathbf{W}\mathbf{H}_B\|_F^2 + \\ &\|\mathbf{V}_C - \mathbf{V}_B^T\mathbf{V}_A^T\mathbf{W}\mathbf{H}_C\|_F^2 + \\ &\|\mathbf{V}_D - \mathbf{V}_C^T\mathbf{V}_B^T\mathbf{V}_A^T\mathbf{W}\mathbf{H}_C\|_F^2 \end{aligned} \tag{26}$$

This following is the pattern for any number of linkages. The pattern is expected to continue beyond 3 linkages i.e. past V_D. [1]

$$\mathbf{W} \leftarrow \mathbf{W} \odot \frac{... + \mathbf{V}_A\mathbf{V}_B\mathbf{V}_C\mathbf{V}_D\mathbf{H}_D^T + \mathbf{V}_A\mathbf{V}_B\mathbf{V}_C\mathbf{H}_C^T + \mathbf{V}_A\mathbf{V}_B\mathbf{H}_B^T + \mathbf{V}_A\mathbf{H}_A^T}{... + \boldsymbol{V}_A\boldsymbol{V}_B\boldsymbol{V}_C\boldsymbol{V}_C^T\boldsymbol{V}_B^T\boldsymbol{V}_A^T\boldsymbol{H}_D\boldsymbol{H}_D^T + \boldsymbol{V}_A\boldsymbol{V}_B\boldsymbol{V}_B^T\boldsymbol{V}_A^T\boldsymbol{H}_C\boldsymbol{H}_C^T + \mathbf{V}_A\mathbf{V}_A^T\mathbf{W}\mathbf{H}_B\mathbf{H}_B^T + \mathbf{W}\mathbf{H}_A\mathbf{H}_A^T} \tag{27}$$

$$\mathbf{H}_A \leftarrow \mathbf{H}_A \odot \frac{\mathbf{W}^T\mathbf{V}_A}{\mathbf{W}^T\mathbf{W}\mathbf{H}_A} \tag{28}$$

$$\mathbf{H}_B \leftarrow \mathbf{H}_B \odot \frac{\mathbf{W}^T\mathbf{V}_A\mathbf{V}_B}{\mathbf{W}^T\mathbf{V}_A\mathbf{V}_A^T\mathbf{W}\mathbf{H}_B} \tag{29}$$

$$\mathbf{H}_C \leftarrow \mathbf{H}_C \odot \frac{\mathbf{W}^T\mathbf{V}_A\mathbf{V}_B\mathbf{V}_C}{\mathbf{W}^T\mathbf{V}_A\mathbf{V}_B\mathbf{V}_B^T\mathbf{V}_A^T\mathbf{W}\mathbf{H}_C} \tag{30}$$

$$\mathbf{H}_D \leftarrow \mathbf{H}_D \odot \frac{\mathbf{W}^T\mathbf{V}_A\mathbf{V}_B\mathbf{V}_C\mathbf{V}_D}{\mathbf{W}^T\mathbf{V}_A\mathbf{V}_B\mathbf{V}_C\mathbf{V}_C^T\mathbf{V}_B^T\mathbf{V}_A^T\mathbf{W}\mathbf{H}_D} \tag{31}$$

If there is a radial branching linkage from matrix V_B that is called V_E then the loss term added would be:

$$\|\mathbf{V}_E - \mathbf{V}_B^T\mathbf{V}_A^T\mathbf{W}\mathbf{H}_E\|_F^2 \tag{32}$$

The chain can then proceed from both V_E and V_C. Every time a branch is encountered a similar term is added.

[1] Experimentally verified to converge for the first linkage

Chained view NMF can be combined with Multiview NMF by creating MUR's according to this equation:

$$\theta \leftarrow \theta * \frac{\nabla_\theta^- f(\theta)}{\nabla_\theta^+ f(\theta)} \tag{33}$$

$\nabla_\theta^+ f(\theta)$ represents the positive terms in the gradient. $\nabla_\theta^- f(\theta)$ represents the negative terms in the gradient.

3 Related Works

In Ailem et al. (2017) we find an approach to encompass a linked dataset as follows, however this is distinct from our approach and does not incorporate multiple chaining. This is formulated identically in Liu (2019) and called multirelational clustering in Equation 34, again this does not incorporate 2nd or beyond chaining of matrices.

$$Loss = 1/2||\mathbf{V} - \mathbf{WH}||_F^2 + 1/2||\mathbf{M} - \mathbf{WSW}^T||_F^2 \tag{34}$$

Where M is the U matrix in SVD for word vectors, S can be an identity matrix.

NMF using MU rules has been an effective method to decompose data matrices for unsupervised feature representations beginning with Seung & Lee (2001). This idea was expanded to multiview NMF that is when there are more than one set of features for a primary key data type Liu et al. (2013). It has been extended to social network community detection as noted in Shi et al. (2015) and Twitter political communities in Ozer et al. (2016). Extending NMF to hypergraphs was done in Wang et al. (2016). Like our approach to find hashtag communities a similar work was performed in He et al. (2017). Shen et al. (2017) analyzes the problem associated with social communities of spammers that spread malicious content using multiview NMF. Yang & Wang (2018) published in IEEE Big Data Mining and Analytics a comprehensive survey on multiview clustering including a section on multiNMF.

The modern trend has been to incorporate deep learning capabilities especially the attention mechanism as seen in, Chen et al. (2019). Additionally the capability to track social media in the recent COVID-19 epidemic was done by Cruickshank & Carley (2020) and could be done to augment the findings in Chen et al. (2020). Applicability of this method is possible from the dataset Chen et al. (2021) in a like manner to Ferrara et al. (2020) or on a related dataset and analysis of Chang et al. (2021).

4 Experiments and Case Studies

4.1 Synthetic Data - Linkages Performance Characteristics and Visualization

The choice of matrix size in the following experiment section is arbitrary, but the rule when linking matrices is the columns of one must correspond to the rows of the next matrix. In practice this is not a problem because if there is a matrix of words $\times$ documents, the matrices that can link are either properties of the documents or the words; for example word vectors or document vectors. We created the first matrix V_A that also can be analyzed alone or linked through the second matrix, V_B, to the third matrix, V_C. This matrix is 630 X 350. The rows represent hashtags and the columns geographic locations.

We also created a second matrix that can be analyzed alone, V_A, or linked to the third matrix, V_B. This matrix has 350 datapoints and 1000 features. We call the features words. So it is basically a graph where each geographic location has predominant words used.

We created a a Gaussian blob dataset with 1000 "words" and 20 features. In this dataset there are 7 groups. This is also called our goal matrix or our third matrix, using the terminology from the equations this is V_C when linked to 2 other matrices. When this matrix has NMF applied to it alone, then it is V_A. Finally, when there is only one other linked matrix this is considered V_B. The 7 groups

Table 1: Synthetic Dataset Performance Characteristics

CHARACTERISTIC	MEAN NMI	STDEV NMI	LINKAGES
Third/Goal Matrix Alone	0.90	0.04	0
Second Matrix Alone*	0.26	0.06	0
First Matrix Alone**	0.11	0.05	0
Second Matrix Chained*	0.49	0.03	1
First Matrix Chained**	0.20	0.04	2

are positive because of the nonnegative requirement imposed by NMF, they are randomly located within the bounding rectangle 1,1 and 40,40.

To give continuity between the matrices we chose 1/7th of each matrix to represent 0,1,2,...,6. So in the first matrix we had 630 hashtags thus there are 90 hashtags per cluster. In the second matrix a likewise grouping was done where 50 rows of locations were in each clustering. Finally in the Gaussian blobs, we have approximately 142 words per cluster.

In the selection of features to represent each of the seven clusters we randomly selected n features that would get a binary value of 1 if they were representative of that group. In essence the first matrix had 50 locations and we chose n randomly for each of the 90 hashtags. n for the first matrix was 14 and 5 for the second matrix. Finally we added random noise at a 5% sparsity to keep the NMI's from being 1.0. This noise is typical in most datasets, else why the need to improve NMI? Likewise, was the choice for n, larger n led to a higher NMI and vice versa. Now it may be asked if information is passing through matrices why not just use the final goal matrix, the reason is because you may want characteristics of the first or second matrix in addition to the goal matrix.

Table 2 contains the NMI or normalized mutual information score for the different number of linkages 0-2. This was provided by the Scikit-Learn module in Python. [2]

4.2 Interpretation of Linked Matrices and Visualization

To understand how information flows through the data tables we illustrated the dot products as they appear in the equations above and plot them in 3 dimensions on a synthetic Gaussian grouping as follows in Figure 4.

4.3 KL Divergence Chained-JAX vs Vanilla KL for 50 Authors

Chained NMF has more usefulness than multiview NMF that cannot handle the case of document$\times$word and word$\times$word vector. We incorporated word vectors with chained view using JAX on the 50 authors C50 dataset. We attained the following results. We did not perform optimization or have any differences between the preprocessing of the two clustering methods. We also did not observe a difference with Frobenius MultiNMF and Vanilla Frobenius NMF which performed better, approx 0.60 NMI and 0.27 ARI. We could optimize TF-IDF values to get similar NMI and ARI values, however then the difference disappeared between KL Vanilla and KL MultiNMF. So in a future work more case studies would be needed. We used the Google 300 dimension word vectors all thresholded at 0 so that the data was non-negative. When investigating the words corresponding to author/topic clusters unfortunately the topics were organized more than the authorship word style and word usage. Only the training half (2500 articles) was used. This dataset was designed not to be topic clustering, however in practice it was.

Additionally, we analyzed the 20 Newsgroups using chained link KL with word vectors vs Vanilla KL and observed a slight decrease in NMI and ARI but better clusters overall found in Table 3.

[2] * and ** are significantly different at a level of 0.001

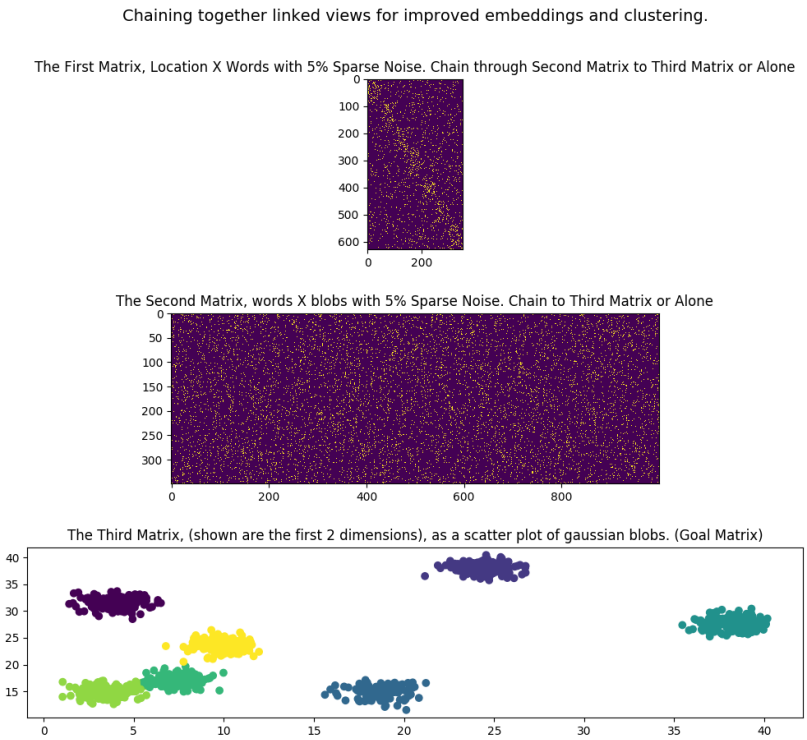

Figure 3: Chaining together a hashtag X location graph to location X word topic graph finally to a word topic X continuous feature blob dataset. Chained views to improve the ability to discern clusters in each of the linking matrices. Shown are the first 2 dimensions of the 20 dimensional dataset.

Table 2: C50 Dataset Performance Characteristics

Cluster Mode and Metric	**RUN 1**	**RUN 2**	**RUN 3**	**RUN 4**	**P-VALUE**
KL Chained-Jax and Word Vectors NMI	0.46	0.45	0.47	0.49	< 0.01
KL Vanilla NMI	0.42	0.42	0.40	0.41	See prior row
KL Chained-Jax and Word Vectors ARI	0.18	0.17	0.18	0.20	< 0.05
KL Vanilla ARI	0.17	0.16	0.14	0.14	See prior row

5 Discussion

The hope of this work is to allow a researcher to collocate all of the relevant data into the model and see what is previously unknown and potentially useful. Their future task will be what a consensus of knowledge can produce, with the hope of a deeper understanding of phenomena that cannot be attained using any single or pair of source data. Artificial intelligence approaches to incorporating a wide range of data has the potential to transform what is considered the lack of algorithmic common sense reasoning to something approaching the ability of human reasoning.

This perspective of multiplying chained matrices together mimics the principle of dilution, for instance if a large matrix multiplies a very small one then most of the information is lost going through the bottleneck. Appropriately, there is nowhere to store the data in the smaller matrix product. Also large matrix to large matrix allows the flow of information readily. And in the third case a small matrix does not have much information to pass through to a larger matrix.

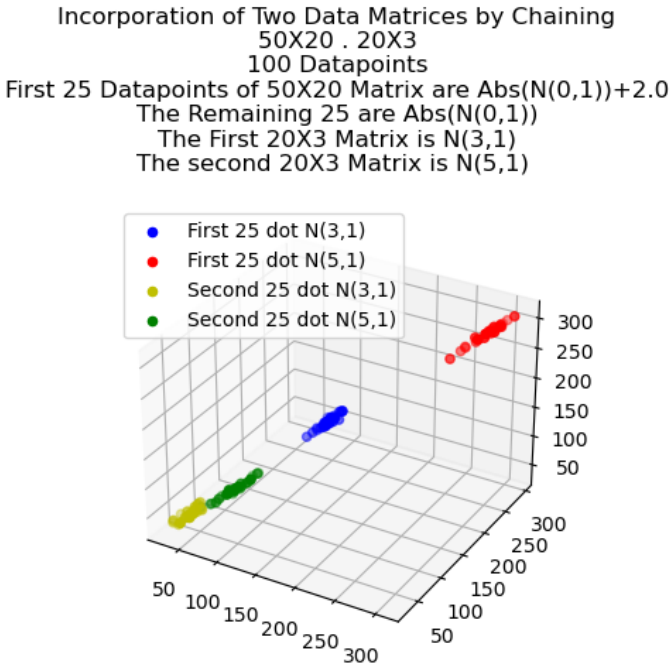

Figure 4: Chaining Visualization on a Controlled Synthetic Dataset.

6 Reproducibility Statement

In the appendix we provide the code used to compare the vanilla NMF to chained link NMF, we however leave the TF-IDF implementation and other details to the reader, for reference we used Scikit-Learn and Python 3.

References

Melissa Ailem, Aghiles Salah, and Mohamed Nadif. Non-negative matrix factorization meets word embedding. In *Proceedings of the 40th international acm sigir conference on research and development in information retrieval*, pp. 1081–1084, 2017.

Juan José Burred. Detailed derivation of multiplicative update rules for nmf. *Paris, France*, 2014.

Ho-Chun Herbert Chang, Samar Haider, and Emilio Ferrara. Digital civic participation and misinformation during the 2020 taiwanese presidential election. *Media and Communication*, 9(1): 144–157, 2021.

Emily Chen, Kristina Lerman, Emilio Ferrara, et al. Tracking social media discourse about the covid-19 pandemic: Development of a public coronavirus twitter data set. *JMIR Public Health and Surveillance*, 6(2):e19273, 2020.

Emily Chen, Ashok Deb, and Emilio Ferrara. # election2020: the first public twitter dataset on the 2020 us presidential election. *Journal of Computational Social Science*, pp. 1–18, 2021.

Hongling Chen, Mingyan Gao, Ying Zhang, Wenbin Liang, and Xianchun Zou. Attention-based multi-nmf deep neural network with multimodality data for breast cancer prognosis model. *BioMed research international*, 2019, 2019.

Iain J Cruickshank and Kathleen M Carley. Characterizing communities of hashtag usage on twitter during the 2020 covid-19 pandemic by multi-view clustering. *Applied Network Science*, 5(1): 1–40, 2020.

Emilio Ferrara, Herbert Chang, Emily Chen, Goran Muric, and Jaimin Patel. Characterizing social media manipulation in the 2020 us presidential election. *First Monday*, 2020.

Chaobo He, Xiang Fei, Hanchao Li, Yong Tang, Hai Liu, and Qimai Chen. A multi-view clustering method for community discovery integrating links and tags. In *2017 IEEE 14th International Conference on e-Business Engineering (ICEBE)*, pp. 23–30. IEEE, 2017.

Jialu Liu, Chi Wang, Jing Gao, and Jiawei Han. Multi-view clustering via joint nonnegative matrix factorization. In *Proceedings of the 2013 SIAM international conference on data mining*, pp. 252–260. SIAM, 2013.

Kai Liu. *A Study of Non-Negative Matrix Factorizations: Foundations, Methods, Algorithms, and Applications*. Colorado School of Mines, 2019.

Mert Ozer, Nyunsu Kim, and Hasan Davulcu. Community detection in political twitter networks using nonnegative matrix factorization methods. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 81–88. IEEE, 2016.

D Seung and L Lee. Algorithms for non-negative matrix factorization. 13:556–562, 2001.

Hua Shen, Fenglong Ma, Xianchao Zhang, Linlin Zong, Xinyue Liu, and Wenxin Liang. Discovering social spammers from multiple views. *Neurocomputing*, 225:49–57, 2017.

Xiaohua Shi, Hongtao Lu, Yangchen He, and Shan He. Community detection in social network with pairwisely constrained symmetric non-negative matrix factorization. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pp. 541–546, 2015.

Wenhong Wang, Yuntao Qian, and Yuan Yan Tang. Hypergraph-regularized sparse nmf for hyperspectral unmixing. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(2):681–694, 2016.

Yan Yang and Hao Wang. Multi-view clustering: A survey. *Big Data Mining and Analytics*, 1(2): 83–107, 2018.

A Appendix

A.1 Caveats

Weakness of this method: The parameters must match exactly to ”chain” together tables. In other words a document X BoW can chain a word X word vector, but if you have a document X BoW + Publication Date, then you must separate the table into two and perform multiview NMF then you can chain together the word vectors. However it should always be possible to include all of the attributes of a relational database to find an embedding of a target feature.

A.2 Proofs and MUR Rules in Code for Reproducing Results

```
import jax.numpy as jnp
from jax import grad

###########Algorithm for Chained Link NMF using KL and Matrix Operations
        for epoch in range(200):
            num = np.divide(Y,np.linalg.multi_dot((X.T,W,H2)))
            num = np.linalg.multi_dot((X,num,H2.T))
            num += np.divide(X,W@H)@H.T
            den = np.ones((W.shape[0],H.shape[1]))@H.T
            den += np.linalg.multi_dot((X,np.ones(Y.shape),H2.T))
            den[den == 0] = eps_machine
            W *= num / den
```

```
            num = W.T@np.divide(X,W@H)
            den = W.T@np.ones((W.shape[0],H.shape[1]))
            den[den == 0] = eps_machine
            H *= num / den

            num = np.divide(Y,np.linalg.multi_dot((X.T,W,H2)))
            num = np.linalg.multi_dot((W.T,X,num))
            den = np.linalg.multi_dot((W.T,X,np.ones((X.shape[1],H2.shape[1]))))
            den[den == 0] = eps_machine
            H2 *= num / den


###########Algorithm for Chained Link NMF using KL and JAX
    t1 = lambda X, W, H: jnp.sum(jnp.multiply(X, jnp.log(jnp.divide(X,W@H))))
    t2 = lambda X: jnp.sum(-X)
    t3 = lambda W, H: jnp.sum(W@H)
    t4 = lambda Y, X, W, H2: jnp.sum(jnp.multiply(Y, jnp.log(jnp.divide(Y,X.T@W@H2)))
    t5 = lambda Y: jnp.sum(-Y)
    t6 = lambda X, W, H2: jnp.sum(X.T@W@H2)

    terms = [t1,t2,t3,t4,t5,t6]
    Args = lambda: [(X,W,H),(X,),(W,H),(Y,X,W,H2),(Y,),(X,W,H2)]
    Windex = [1,None,0,2,None,1]
    Hindex = [2,None,1,None,None,None]
    H2index = [None,None,None,3,None,2]


     MU(Xindex):
        args = Args()
        grads = [grad(t,argnums=i)(*a) for a,i,t in zip(args,Xindex,terms) if i is no
        numerator = [np.abs(g) for g in grads if np.any(g < 0)]
        numerator = np.sum(numerator,axis=0)
        denominator = [g for g in grads if np.any(g > 0)]
        denominator = np.sum(denominator,axis=0)
        return  numerator / (denominator + eps_machine)

    for epoch in range(200):
        W *= MU(Windex)
        H *= MU(Hindex)
        H2 *= MU(H2index)

###########Algorithm for Chained Link NMF using Frobenius and
###########JAX or Matrix Operations
        '''
        min ||X-WH|| + ||Y-XTWH2|| s.t. W, H, H2 >= 0

        tr((X-WH)T (X-WH))
        tr((Y-XTWH2)T (Y-XTWH2))
        '''
        t1 = lambda X: jnp.trace(X.T@X)
        t2 = lambda X, W, H: -jnp.trace(X.T@W@H)
        t3 = lambda X, W, H: -jnp.trace(H.T@W.T@X)
        t4 = lambda W, H: jnp.trace(H.T@W.T@W@H)

        t5 = lambda Y: jnp.trace(Y.T@Y)
        t6 = lambda Y, X, W, H2: -jnp.trace(Y.T@X.T@W@H2)
        t7 = lambda Y, X, W, H2: -jnp.trace(H2.T@W.T@X@Y)
        t8 = lambda X, W, H2: jnp.trace(H2.T@W.T@X@X.T@W@H2)
```

```
        for epoch in range(200):
            Wnum = X@H.T + np.linalg.multi_dot([X,X.T,W,H2,H2.T])
            Wden = np.linalg.multi_dot([W,H,H.T]) + np.linalg.multi_dot([X,Y,H2.T])
            Wden[Wden==0] = eps_machine
            #W *= MU(Windex)
            W *= Wnum/Wden

            Hnum = W.T@X
            Hden = np.linalg.multi_dot([W.T,W,H])
            Hden[Hden == 0] = eps_machine
            H *= Hnum/Hden
            #H *= MU(Hindex)

            H2num = np.linalg.multi_dot([W.T,X,Y])
            H2den = np.linalg.multi_dot([W.T,X,X.T,W,H2])
            H2den[H2den == 0] = eps_machine
            u1 = H2num / H2den
            #u2 = MU(H2index)

##Basic Procedure for Testing AutoDiff Gradient Matches Matrix Operation
##(Proof of Correctness)

from jax import grad
import jax.numpy as jnp
import numpy as np
X=np.random.normal(0,1,(15,20))
Y=np.random.normal(0,1,(20,25))
W=np.random.normal(0,1,(15,7))
H=np.random.normal(0,1,(7,20))
H2=np.random.normal(0,1,(7,25))
X,Y,W,H,H2 = [np.abs(x) for x in [X,Y,W,H,H2]]
t4 = lambda Y, X, W, H2: jnp.sum(jnp.multiply(Y, jnp.log(jnp.divide(Y,X.T@W@H2))))
g1 = grad(t4,argnums=3)(Y,X,W,H2)
num = np.divide(Y,np.linalg.multi_dot((X.T,W,H2)))
num = np.linalg.multi_dot((W.T,X,num))
g2 = num
if np.allclose(g1,-g2):
    print("H2 Num is good.")
```

Table 3: 7 Newsgroups Characteristic Cluster Comparison (subset of 20 Newsgroups)

	Chained Link NMF KL 7 Cluster Word Choices for 7 Newsgroups
talk.politics.mideast	israel, israeli, government, said, state, palestinian, part, arab, war, during, peace, against, back, year, children, off, agree, soldiers, american, troops
sci.electronics	power, circuit, battery, edu, voltage, light, space, current, radio, sky, electronics, output, thanks, 10, energy, information, msg, tv, off, temperature
sci.med	she, her, medical, doctor, disease, pain, patients, treatment, cancer, go, food, cause, drug, medicine, ago, patient, really, effects, problem, wife
sci.space	space, launch, ground, money, msg, nasa, cost, moon, high, radar, re, low, costs, dc, funding, still, safety, buy, long, real
sci.crypt	key, chip, clipper, encryption, government, phone, keys, system, nsa, algorithm, law, security, escrow, des, number, public, crypto, data, bit, secure
comp.graphics	thanks, graphics, program, please, image, file, files, software, mail, looking, ftp, space, format, hi, advance, images, help, gif, pc, windows
soc.religion.christian	god, jesus, church, jews, christian, believe, christ, christians, bible, israel, him, armenian, faith, jewish, armenians, sin, world, life, turkish, christianity
	Vanilla NMF KL 7 Cluster Word Choices for 7 Newsgroups
talk.politics.mideast	1993, during, april, armenians, argic, 10, armenian, between, 93, ground, 000, center, last, 12, 30, 19, against, city, air, russian
sci.electronics	better, company, around, few, enough, actually, buy, back, going, re, cost, big, 10, off, build, go, long, probably, got, down
sci.med	edu, cause, banks, back, case, cadre, doctor, effect, ago, gordon, medical, chastity, disease, help, soon, anything, deleted, intellect, day, geb
sci.space	thanks, please, mail, advance, space, email, book, nasa, looking, another, information, anybody, articles, look, ll, contact, able, around, copy, help
sci.crypt	clipper, chip, key, government, encryption, system, algorithm, public, keys, data, com, bit, phone, nsa, security, using, chips, secret, already, anything
comp.graphics	thanks, graphics, file, available, looking, please, files, ftp, program, image, advance, help, hi, appreciated, 3d, bit, code, format, etc, color
soc.religion.christian	god, believe, christian, israel, him, church, again, against, come, both, fact, christians, christ, between, agree, another, bible, said, jesus, however