Rank Selection in Non-negative Matrix Factorization using Minimum Description Length

Steven Squires [1]

[1]Electronics and Computer Science, University of Southampton.

Adam Prügel-Bennett [1]

[1]Electronics and Computer Science, University of Southampton.

Mahesan Niranjan [1]

[1]Electronics and Computer Science, University of Southampton.

Keywords: Non-negative matrix factorization, minimum description length, rank selection, subspace size selection

Abstract

Non-negative matrix factorisation (NMF) is primarily a linear dimensionality reduction technique that factorizes a non-negative data matrix into two smaller non-negative matrices: one that represents the basis of the new subspace and the second that holds

the coefficients of all the data-points in that new space. In principle, the non-negativity constraint forces the representation to be sparse and parts based. Instead of extracting holistic features from the data, real parts are extracted that should be significantly easier to interpret and analyse. The size of the new subspace selects how many features will be extracted from the data. An effective choice should minimise the noise whilst extracting the key features. We propose a mechanism for selecting the subspace size by using a minimum description length technique. We demonstrate that our technique provides plausible estimates for real data as well as accurately predicting the known size of synthetic data. We provide an implementation of our code in a Matlab format.

1 Introduction

1.1 Non-negative Matrix Factorization

Consider a data matrix $\mathbf{V} \in \mathbb{R}^{m \times n}$ with m dimensions and n data points which has only non-negative elements. If we define two matrices, also with only non-negative elements: $\mathbf{W} \in \mathbb{R}^{m \times r}$ and $\mathbf{H} \in \mathbb{R}^{r \times n}$, then non-negative matrix factorisation (NMF) can reduce the dimensionality of $\mathbf{V}$ through the approximation:

$$\mathbf{V} \approx \mathbf{WH} \tag{1}$$

where, generally, $r < n$ and $r < m$.

The columns of $\mathbf{W}$ make up the new basis directions of the dimensions we are projecting onto. Each column of $\mathbf{H}$ represents the coefficients of each data point in this new subspace.

There has been a considerable increase in interest in NMF since the publication of a seminal work by Lee and Seung (1999) in part because NMF tends to, naturally, produce a sparse and parts based representation of the data. This sparse and parts based representation is in contrast to other dimensionality reduction techniques such as principal component analysis which tends to produce a holistic representation. The parts should represent features of the data, therefore NMF can produce a representation of the data by the addition of extracted features. This representation may be considerably more interpretable than more holistic approaches.

There are a range of algorithms to conduct NMF most of them involving minimising an objective function such as:

$$\min ||\mathbf{V} - \mathbf{WH}||^2_{\text{Fro}} \text{ subject to } W_{i,j} \geq 0, H_{i,j} \geq 0.$$

Other objective functions such as the Kullbeck-Leibler divergence are also options. All the popular algorithms rely on a pre-chosen value of r, the size of the new subspace. In NMF the size of the subspace has real meaning: it selects the number of features extracted. If r is chosen too low we are likely to miss features and if r is chosen too large then we will probably model noise. A good choice of r then reduces the noise in the data whilst effectively modelling the key features.

1.2 Rank Selection in Non-negative Matrix Factorization

In a recent review Gillis (2014) put forward three main methods for selection of r: use of expert insight; trial and error; and use of singular value decomposition. Expert insights are invaluable but suffer for three main reasons: 1) there may be no expert capable of selecting a good choice of r; 2) the experts may select r incorrectly; 3) even

if an expert is able to effectively select r then independent confirmation is useful to add weight to the expert opinion.

Trial and error in this context means trying different values of r and then manual selection of one that best fits the aim of the researcher for that particular application. This method suffers as it is hard to know what a "good" solution looks like. Trial and error can be dangerous in that it allows researchers to tune their results in a manner which produces the solution best for their work, so that a "good" solution becomes the solution that confirms their hypothesis.

Singular value decomposition is applied by selecting r when the values of the singular values becomes "small". The challenge is that unless there is a clear fall towards zero the choice of where the values become "small" is very difficult to make.

There are several more involved methods that have been proposed for the selection of the rank. Examples include the use of cross-validation (Kanagal and Sindhwani , 2010; Owen and Perry , 2009) and the use of Stein's unbiased risk estimator (Ulfarsson and Solo , 2013). Cross-validation, in particular, is a common technique across supervised learning for assessing the quality of a model. In NMF, an unsupervised model, cross-validation essentially requires the imputation of missing data. There are different techniques for achieving this and Kanagal and Sindhwani (2010) showed that these different techniques can produce significantly different estimates of r or sometimes no estimate at all.

There is also an approach to NMF by Blei (2010) using a Bayesian formulation which offers the benefit of selecting r whilst finding $\mathbf{W}$ and $\mathbf{H}$. They impose a prior belief that the rank should be small and from there find a solution which fulfils this

prior but this requires domain expertise to determine the choice of a good prior. Our aim is to offer our approach as an additional method to help to guide a choice of r for researchers using NMF, in particular when there is little or no domain knowledge available.

1.3 Approach and Contribution

Our approach is to utilise a minimum description length (MDL) technique to find the best trade-off between a low r which misses key features and a high r which models noise. We suggest a pair of methods for applying MDL to NMF to assess the best choice of r. Our algorithms, which are available in Matlab format, allow for the estimation of the best value of r and can produce a range of graphs that can be used to analyse the quality of the estimation.

In the next section we will introduce the background and theory behind MDL, we then will propose our solution to find the minimum description length. In Section 3 we apply our MDL technique to real and synthetic data demonstrating the validity of the technique. Finally, in Section 4 we discuss the results we obtained and explain why we believe our technique is a useful addition to the NMF toolbox.

2 Minimum Description Length

2.1 Background and theory

Minimum description length (MDL) is a method for selecting between models of varying complexity. At its core is the idea that the best model is one that compresses the

data most effectively. As the best way of compressing the data would also involve the smallest transmission cost when sending an encoded message, compression of the data and transmitting the shortest message are essentially equivalent.

In the NMF case the message is the matrix $\mathbf{V}$ which is approximated using $\mathbf{WH}$. The model is simple when r is small and, consequently, $\mathbf{W}$ and $\mathbf{H}$ have few elements which are cheap to encode. However, with a small r the approximation $\mathbf{WH} \approx \mathbf{V}$ is likely to be poor, requiring an addition to the message to correct the poor approximation. The MDL principle is to choose the model that minimises the total message length (Wallace and Boulton , 1968). By trading off between the complexity and accuracy of the model, we hope to find the level of complexity which minimises the transmission of noise whilst maximising the transmission of real features.

There needs to be pre-agreement between the message transmitter and receiver about the level of precision, $\delta\mathcal{D}$, that the data, $\mathcal{D}$, should be set to. The message must be communicated to this agreed precision. This means the message will consist of the model, $\mathcal{H}$, and corrections to the model to reproduce the original data matrix exactly. Therefore the message length, $L(\mathcal{D}, \mathcal{H})$, consists of two parts (MacKay , 2003):

$$L(\mathcal{D}, \mathcal{H}) = L(\mathcal{H}) + L(\mathcal{D}|\mathcal{H})$$

where $L(\mathcal{H})$ is the length of the hypothesis, or the complexity of the model, and $L(\mathcal{D}|\mathcal{H})$ encodes the accuracy of the model. More complex models will tend to have a larger $L(\mathcal{H})$ and a smaller $L(\mathcal{D}|\mathcal{H})$.

Two important points should be made here. First we are not interested in how to actually optimally encode the message, we are only interested in the message length

itself. Secondly, for model selection we are only interested in the relative length of each message. Any additional pieces of information required in the message that are consistent across all the different values of r are irrelevant in MDL because they will increase the total description length by a constant amount and make no difference to the location of the minimum. The only terms that matter are those that will be different across different values of r. In other words, we are not interested in the message itself, or the absolute cost of encoding the message, but in the relative cost of sending the message at different values of r.

2.2 Proposed MDL Algorithm

To perform MDL to assess an appropriate subspace size for NMF we first must specify the components of $L(\mathcal{H})$ and $L(\mathcal{D}|\mathcal{H})$. The encoded length of the hypothesis, or the complexity of the model, $L(\mathcal{H})$ is $L(\mathbf{W}) + L(\mathbf{H})$ where $L(\mathbf{W})$ and $L(\mathbf{H})$ are the length of messages required to encode the matrices $\mathbf{W}$ and $\mathbf{H}$ respectively. The $L(\mathcal{D}|\mathcal{H})$ term is the length of the correction required to ensure that $\mathbf{V}$ can be reproduced exactly (to pre-specified precision) and is the encoded length of the matrix of errors, $L(\mathbf{E})$, where $\mathbf{E} = \mathbf{V} - \mathbf{WH}$. Implementation of MDL then requires the estimation of the minimum length of code that would allow the three matrices $\mathbf{E}$, $\mathbf{W}$ and $\mathbf{H}$ to be encoded into a message. When r is small the matrices $\mathbf{W}$ and $\mathbf{H}$ are small and so cost relatively little to encode, but the error matrix, $\mathbf{E}$, is large and therefore expensive to encode. As we increase r the errors reduce so the cost of transmitting $\mathbf{E}$ falls, but the cost of transmitting the model, $\mathbf{W}$ and $\mathbf{H}$, increases. At some point there should be an r value at which the total length is minimised, this is then the minimum description length and

gives us a choice for r.

The principle of MDL relies on the use of the best possible encoding of the data, that is the encoding with the lowest cost. An upper bound on potential encodings can be estimated by considering the information content of each element. In general, any value which occurs multiple times is cheap to encode. To understand why, assume that the values in the error matrix are Gaussian distributed, then many elements will fall into a range that is close to the mean which can be assigned a short code. Any element far from the mean will require a longer code and is therefore more expensive to send. The Shannon information content allows us to estimate this cost using probabilities and is defined as (MacKay , 2003):

$$h(x) = -\log_2 P(x)$$

where x is the value of an element and $P(x)$ the probability of that value occurring. The aim is to find the probability of a value occurring in the **W**, **H** and **E** matrices then to convert that value to a cost using the Shannon information content.

To estimate the probabilities we separate the data into bins of width $\delta\mathcal{D}$, which is the precision of the data. This value should be assessed from the data itself. We then apply two methods to estimate the probability of a term occurring in that bin. The first is to use the frequency of terms in that bin, n_i, compared to the total number of terms, N, so that $P(x) = \frac{n_i}{N}$ where $P(x)$ is the probability of an element x to be in the i^{th} bin. There is a considerable problem with this method in that while we can estimate the probabilities of each element in each bin, and hence the bound on the cost of sending the data, we also would need to send a specification of the histograms themselves, essentially the

starting and end points of the histograms, along with the code used for each histogram. The bin width could be assumed to be the precision. The encoding of starting and end points of the histogram are likely to be fairly similar across r and there should be fairly inexpensive methods of encoding which bins are assigned to which codes but it is not a trivial task to complete. It is, however, likely that the parameters of this histogram model will be dwarfed by the cost of encoding the data itself. In the rest of this report we will refer to this technique as the histogram method.

The second method of estimating probabilities is more consistent with MDL principles but also suffers from a potential problem. Instead of using the frequencies of the histograms themselves, probability distributions are applied to the binned data, which allows us to find the probability density, ρ_i of each bin, i. The probability for an element, x, in the i^{th} bin is then $P(x) = (\rho_i \times \delta\mathcal{D})$. The advantage over the previous method is that the technique required to send the message is quite straightforward. As long as we use fairly simple distributions we must simply encode the parameters of the model and send them. The receiver can then recreate the distributions and will therefore be able to recreate the message. The only change in the model as r changes will be in the few parameters of the distribution (for a Gaussian distribution the mean and standard deviation, for example) which is highly unlikely to have any noticeable effect on the description lengths for any reasonably large data matrix. The potential problem with this method is that if the distributions do not fit well with the data, the estimates of the probabilities will not be accurate. This will then overestimate the description length.

The probability distributions to fit the non-zero terms from $\mathbf{W}$ and $\mathbf{H}$ should possess some features: it must be non-negative therefore the probability density should tend

towards zero or infinity at zero values and the probability density should tend towards zero as the value becomes large. A simple choice is the gamma distribution which has a probability density function (PDF) of:

$$\rho(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{(\alpha-1)} e^{-\beta x}$$

where Γ is the gamma function, α and β are parameters. This is quite a flexible family of distributions that is often able to approximate real world distributions quite accurately.

At this point there is both a challenge and an opportunity. Part of the value of NMF is that it naturally tends to result in sparse matrices with a relatively high proportion of zero terms. The opportunity is that these zero terms could be sent very cheaply as separate matrices and the challenge is that these zero terms may result in highly inaccurate distributions being set to the **W** and **H** terms. The PDF of the gamma distribution either falls to zero or tends to infinity at zero depending on the parameter α. If the non-zero data is best fit by a distribution which tends to zero the estimates of the probabilities will be very poor. Most seriously they might well significantly overestimate the cost of sending the zero terms. It may be better to split the data up into zero-terms and non-zero terms. The separation of zero and non-zero terms requires some threshold to be set. Above the threshold the data is modelled using a gamma distribution and below the threshold the data is separately encoded, as described below. The Matlab code we provide allows for the manual choice of the threshold but also for an automatic choice made by applying MDL techniques themselves.

The automatic threshold is selected by systematically searching through the space of zero-thresholds for both **W** and **H** from zero up to the edge of the first bin. The total

description length is then calculated and the lowest value is selected. This can result in different thresholds for **W** and **H** and also across different r terms.

The matrices containing the zero terms, $\mathbf{W}_0$ and $\mathbf{H}_0$, are encoded via the probability of being a zero which is given by n_0/n_T where n_0 is the number of zero values and n_T is the total number in **W** or **H**. This leads to:

$$L(\mathbf{X}_0) = -n_0 \log_2 \frac{n_0}{n_T} - (n_T - n_0) \log_2 \left(\frac{n_T - n_0}{n_T} \right)$$

where $\mathbf{X}_0$ represents either $\mathbf{W}_0$ or $\mathbf{H}_0$ and the n terms are the numbers for $\mathbf{W}_0$ or $\mathbf{H}_0$ respectively. The second term encodes the cost of specifying the terms that are non-zero. This can be viewed as sending a code specifying a matrix of zeros and ones followed by the distribution and the codes for the non-zero terms.

This separation of the **W** and **H** matrices results in a total description length of:

$$L(\mathcal{D}, \mathcal{H}) = L(\mathbf{W}_0) + L(\mathbf{W}_+) + L(\mathbf{H}_0) + L(\mathbf{H}_+) + L(\mathbf{E}) \tag{2}$$

where $L(\mathbf{W}_0)$, $L(\mathbf{H}_0)$ are the description lengths required to encode the zeros in the **W** and **H** matrices respectively; $L(\mathbf{W}_+)$, $L(\mathbf{H}_+)$ are the description lengths required to encode the non-zero terms in the **W** and **H** matrices respectively; and $L(\mathbf{E})$ is the description length to encode the error terms.

The non-zero data is assigned to bins of width $\delta\mathcal{D}$ and a gamma distribution is separately fitted to the $\mathbf{W}_+$ and $\mathbf{H}_+$ data. The probabilities, followed by the Shannon information content and hence the description lengths are then calculated. The **W** and **H** matrices that would be found from the message are calculated followed by the error matrix **E**. A Gaussian probability distribution is set to the error matrix to enable the

extraction of the probabilities and description lengths. The five terms that make up the description length are summed to give the total description length as in Eq. (2). Our technique is explained in Algorithm 1.

Algorithm 1 MDL algorithm for each r value with automatic moving zero threshold

Input: **V**, **W**, **H**, δD

Output: Description lengths for each r

1: **for** zero threshold values of **W** and **H**

2: Separate out zero values, calculate $L(\mathbf{W}_0)$ and $L(\mathbf{H}_0)$

3: Apply gamma distributions to $\mathbf{W}_+$ and $\mathbf{H}_+$, calculate $L(\mathbf{W}_+)$ and $L(\mathbf{H}_+)$

4: Calculate **E** then $L(\mathbf{E})$

5: Calculate $L(\mathcal{D}, \mathcal{H})$

6: **if** $L(\mathcal{D}, \mathcal{H})$ is smaller than previous smallest, **then** store description lengths **endif**

7: **end for**

8: Return $L(\mathcal{D}, \mathcal{H})$

3 Application of Minimum Description Length

To demonstrate the application of our MDL technique we have applied it to real and synthetic datasets. The results shown in this paper utilise the NMF method of Hoyer (2004) without additional sparseness constraints added (but we also tested other methods taken from (Gillis , 2014) and see no notable differences). It is important to emphasise that there is no real ground truth in the real data we assess, so we cannot demonstrate beyond reasonable doubt that our technique works effectively on real data. However, there are several criteria we would expect our method to meet if it is capable of selecting an

appropriate r:

1. That the MDL technique performs in the manner we anticipate, i.e. that $L(\mathcal{D}|\mathcal{H})$ would fall and $L(\mathcal{H})$ should rise as r increases, and there should be a turning point in $L(\mathcal{D}, \mathcal{H})$.

2. That the MDL technique picks a plausible value of r for real data, especially if this is similar to choices made using other methods, such as use of external knowledge.

3. That the MDL technique can reasonably estimate r-values from synthetic data with a known r.

4. That MDL shows clear estimates of r for different types of data.

5. That the choice of r is robust to some variation in the data.

In Figure 1 we demonstrate the success of MDL in achieving the first and second points and part of the fifth. The left plot shows real data of a set of 2429 images of faces (see Table 1) with 361 dimensions (pixels) used by Lee and Seung (1999). The description lengths change exactly as we would expect: the length of the errors falls with increasing r, at the same time the $L(\mathbf{W})$ and $L(\mathbf{H})$ terms grow larger. The MDL algorithm produces exactly the pattern that we would expect, fulfilling our first criterion. This same plot also demonstrates that MDL can meet the second criterion, the straight line down to $r = 80$ shows the r-value of the minimum description, but the turning point is fairly flat and a reasonable choice could be anywhere from $r = 50$ to $r = 100$. Here we should note that when Lee and Seung Lee and Seung (1999) used this data-set

they chose $r = 49$ for their subspace size. They do not specify how they chose r but it may well have been via a trial and error approach choosing that value when it gave good plots for their paper. Using MDL we have found a result in a similar range with no parameter tuning or assumptions beyond the choice of precision. This estimated value of r is certainly a sensible value and the turning point is clear. We have also included results from re-running the NMF algorithm on the data, which show no difference in the choice of r and produce virtually identical lengths, in fact the differences in the results are difficult to spot in the figure. This identical solution to re-runs of the data is significant as NMF does not necessarily produce one unique solution. Instead all the re-runs of the algorithm are likely to have produced somewhat different $\mathbf{W}$ and $\mathbf{H}$ matrices. Our estimation of r does not change at all implying a level of consistency across different NMF solutions. A final point to be noted from this plot is that the solid line shows results from the distributions while the dashed line shows the results from using the histograms alone. There is no difference in estimation of r and only small differences in description length values between the two methods. As both methods have potential, but complementary, flaws, the similarity in output implies that these flaws do not adversely affect the conclusion.

The right plot in Figure 1 shows MDL applied to synthetic data with $m = 1000$ and $n = 2000$. This simple synthetic data is created by creating two matrices $\mathbf{W} \in \mathbb{R}^{m \times r}$ and $\mathbf{H} \in \mathbb{R}^{r \times n}$ with random locations of random non-zero terms. These are multiplied together and additional noise added. The size of the subspace r is 150 and is estimated correctly by the MDL approach. Clearly this data is simple and the selection of an appropriate r from here does not prove our approach is effective but it does show that

Table 1: Data-sets names, the type of data, the number of dimensions, m, and number of data-points, n.

Name	**Type**	m	n	**Source**
Faces	Image	361	2429	http://cbcl.mit.edu/software-datasets /FaceData2.html
Genes	Biological	5000	38	http://www.broadinstitute.org/cgi-bin /cancer/datasets.cgi
FTSE 100	Financial	1305	94	University of Southampton Bloomberg information terminal

MDL can find the appropriate value of r for some datasets. Again there is no difference in the conclusions drawn from the histograms and the distributions. We thus claim that our MDL algorithms can fulfil the third criteria we set out.

The left plot in Figure 2 shows the total description lengths for several different data types (see Table 1) and allows us to meet our second and fourth criteria. There is no real ground-truth to these datasets so it is not possible to confirm that MDL is picking a good choice of r. It is, though, selecting an r that seems to be reasonable for each of the plots and also different from each other. If we were seeing all the turning points at similar values we might suspect that it was a feature of the algorithm rather than the data, the different locations of the turning points suggests that it is extracting information from the data itself. The Genes dataset has been extensively used, often with an implied r of 2 or 3 (Devarajan , 2008) which is similar to our estimate of between 2 and 5. Our estimate of the FTSE 100 dataset r-value is around $r = 8$, where the value of r could

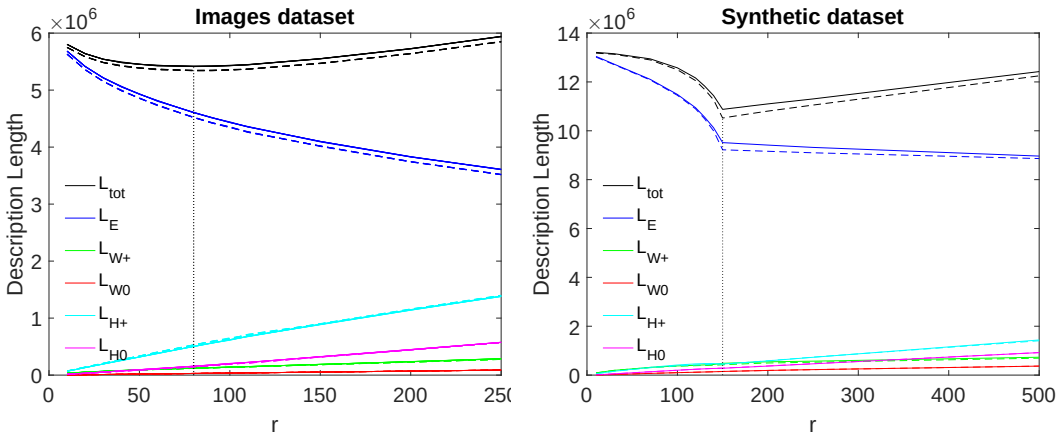

Figure 1: Left) The description lengths for the Faces dataset, showing a minimum of the total description length at around $r = 80$, with a reasonable range from $r = 50$ to $r = 100$. The solid line is for the description lengths found using the distributions and the dashed line from the histograms, the choice of r is the same. Right) The description lengths for synthetic data with a real r of 150. MDL shows a clear minimum at $r = 150$, perfectly estimating the correct value.

be considered as the number of economic sectors, such as energy, telecommunications, IT etc. An estimate of the number of these sectors of around ten would be reasonable, and is close to our evaluation.

The right plot of Figure 2 shows a range of results for synthetic data created as discussed earlier but with r values of 25, 50, 80, 120 and 150. The black vertical lines show the actual location of the real r value. For all results except for $r = 25$ the MDL estimation is identical to the actual value. Our algorithm is correctly estimating the real value of r.

The final aspect of our technique we will consider is the robustness of our technique to certain changes. We have already demonstrated that our technique is robust to re-runs

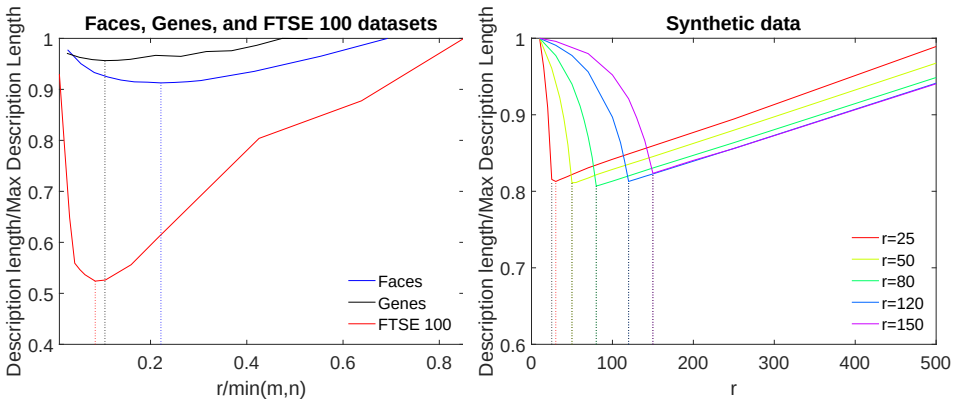

Figure 2: Left) The total description length for a range of datasets showing where turning points occur, potentially signalling an effective choice of r. While no ground truth is known the plots show minima at sensible locations, which correspond reasonably well with estimates from other sources. Right) The total description length for a range of synthetic data, MDL correctly identifies the r term for each dataset except for $r = 25$ which is still very close (the red vertical line) to the correct value (the black line).

of the NMF algorithm which can produce significantly different **W** and **H** matrices. To further attempt to get an impression of the uncertainty in our technique we applied bootstrapping to the faces dataset to produce five different variations of the original, in addition to the non-bootstrapped variant. NMF was then used to find the **W** and **H** matrices and our MDL technique applied. In the left plot of Figure 3 we see the results from applying the MDL techniques to this dataset. The solid line shows the results of applying MDL using the distributions for the images of faces dataset. The dotted line shows the same but for bootstrapped data, this is hard to see as the results are almost identical. The dashed line shows the same but for MDL applied using the histograms and the dash-dot line for the equivalent bootstrapped results. The differences are very

marginal and the choice of r is similar for both. There is almost no difference seen in results when bootstrapping the data, we therefore consider the method to be reasonably robust to this type of alteration of the data.

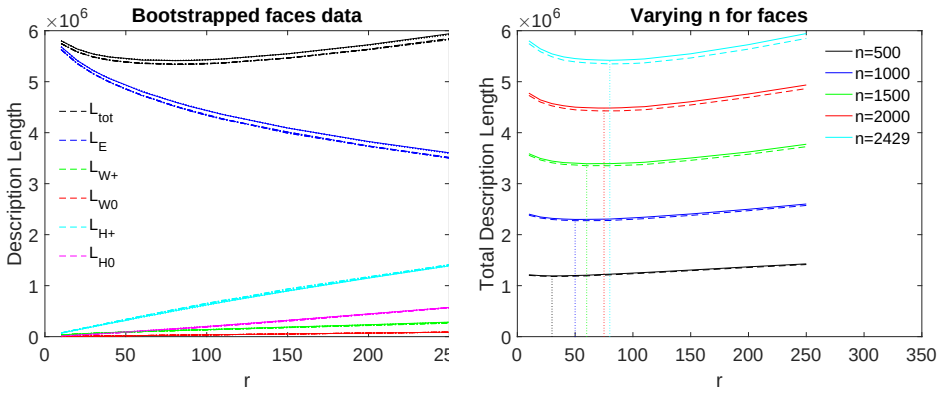

Figure 3: Left) The solid line shows the results of applying MDL using the distributions for the images of faces dataset. The dotted line shows the same but for bootstrapped data, these are hard to see as the results are almost identical. The dashed line shows the same but for MDL applied using the histograms and the dash-dot line for the equivalent bootstrapped results. Again there is almost no difference, our results show no significant variation under bootstrapping. Right) The results of L_{tot} for varying reduced size of n for the images of faces dataset. We see a clear reduction in the optimal r value as n is reduced. The dashed line is for the histogram plots and the solid line for the distributions.

The right hand plot of Figure 3 shows how the location of the MDL selection of r changes with the number of samples from $n = 500$ up to $n = 2429$ (the full dataset). The vertical lines record the value of the minimum for each sample size. It is apparent that the choice of r decreases with a smaller sample size. The final two terms with

$n = 2000$ and $n = 2429$ are similar, but there is a considerable fall in the selected r-value with smaller n. We offer an explanation of why we see such a fall in the best choice of r consistent with these results. If we consider the data to be made up of features with a range of importance, in the case of a set of images of faces important features might be eyes, noses, ears or mouths. These features, and variants of them, will be required for almost all faces. On the other hand features such as moustaches are far less common. With lower numbers of samples it may be better to assume a moustache feature, which may be used by a small number of images, is not worth considering as a feature, instead a moustache can be considered as noise and accepted as part of the corrections made by the $\mathbf{E}$ matrix. As the number of samples increases it becomes possible to recognise that the extra feature is not noise and so the number of features expand, the capacity of the model increases with more data. We can, potentially, see the features that appear at low n as the more important features and as n increases we gain the features that are either less important to much of the data or important to only a small subset of the data. In reality this analysis of the data may be overly simplistic, in that the smaller number of features are likely to partially include the less important features, we may well then see combined features rather than the less relevant features being completely absent. Either way, as the number of data points increases so does the capacity of the model.

4 Conclusion

Our novel technique is to apply an MDL technique to selection of r. Before considering the results there are several attractive features of MDL. First, all the data is used in MDL, there is no need to keep hold-out folds so no need to average out the results from the different folds or to consider the variance in the results when drawing your conclusions, as there is in techniques such as cross-validation. Second, MDL is an elegant technique with intuitive appeal which gives a natural trade-off between errors and model size. Third, the only potentially arbitrary parameter is precision, δD, but this has only a minor influence on the relative description length of different models and, in any case, may not be arbitrary if the precision of the data itself can be used.

We have applied our MDL technique to a range of real and synthetic data. MDL is able to accurately estimate r in synthetic data as well as providing reasonable estimates of the best r in real data. Our technique is robust to re-runs of the NMF algorithm and to bootstrapping the data, producing the same predictions of the best r.

Our technique has been tested on a range of data and is likely to work better on some data than others. If the distributions of the matrices **W**, **H** and **E** match our set distributions well then we would expect to make good predictions. Conversely if the distributions do not match the data well our estimates may be inaccurate. In particular, if **V** is highly sparse, most of the errors will probably be zero and our algorithm may require some alteration. An advantage of our algorithms is that problems should be observed in differences in results from the histogram and gamma distribution methods. Our algorithms also allow for the production of a range of graphs to test the similarity of distributions to the actual data, which should highlight potential problems. Extensions

to this work would likely be to investigate whether other distributions do a better job than our Gaussian and gamma choices.

There are a range of techniques for assessing an appropriate value of r in the literature. The best method is likely to utilise several techniques to select an appropriate r. We would suggest our technique adds to the potential toolbox that NMF researchers utilise to form judgements about the choice of r.

References

Blei, David M & Cook, Perry R & Hoffman, Matthew (2010). Bayesian nonparametric matrix factorization for recorded music. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 439–446,

Devarajan, Karthik (2008). Nonnegative matrix factorization: an analytical and interpretive tool in computational biology. *PLoS Comput Biol*, *4*,

Gillis, Nicolas. The why and how of nonnegative matrix factorization. *Regularization, Optimization, Kernels, and Support Vector Machines*, *12*, 257 – 291.

Hoyer, Patrik O. Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research*, *5*, 1457–1469.

Kanagal, Bhargav & Sindhwani, Vikas (2010). Rank selection in low-rank matrix approximations: A study of cross-validation for NMFs. *IN: Advances in Neural Information Processing Systems (NIPS)*.

Lee, Daniel D. & Seung, H Sebastian (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, *401*, 788 – 791.

MacKay, David JC (2003). Information theory, inference and learning algorithms. *Cambridge university press*.

Owen, Art B & Perry, Patrick O (2009). Bi-cross-validation of the SVD and the non-negative matrix factorization. *The annals of applied statistics*, 564–594.

Ulfarsson, Magnus O & Solo, Victor (2013). Tuning parameter selection for nonnegative matrix factorization. *ICASSP*, 6590–6594.

Wallace, Christopher S & Boulton, David M (1968). An information measure for classification. *The Computer Journal*, *11*, 185 – 194.