The Pennsylvania State University
The Graduate School

PARSIMONIOUS MODEL SELECTION BASED ON BAYESIAN INFORMATION CRITERION

A MIXTURE BASED HYBRID MODEL ON IMAGE CLASSIFICATION

A Thesis in
Electrical Engineering
by
Hang Wang

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2019
The thesis of Hang Wang was reviewed and approved* by the following:

David J Miller  
Professor  
Thesis Advisor

Constantino Lagoa  
Professor

Kultegin Aydin  
Professor  
Head of the Department of Electrical Engineering and Computer Science

*Signatures are on file in the Graduate School
ABSTRACT

My thesis research contains two parts of work. The first part of my work revisits a previous paper about Parsimonious Topic Model, and modifies the Bayesian Information Criterion (BIC) given in this paper. By giving a cheaper cost representation of the model structure, we got a more sparse topic model, which is also better than the previous model on fit goodness, and label consistency. The second part of my work is about a hybrid model which makes use of both Transfer Learning and Convolutional Neural Network. We found that, with very limited dataset, Transfer Learning does a better job because it doesn’t need too many parameters; when we have a large training set so that we can train a very sophisticated model, Convolutional Neural Network can do a better job; with a training of middle size, our hybrid model can give best classification accuracy.
# TABLE OF CONTENTS

LIST OF FIGURES .................................................................................................................. v

Chapter 1  Parsimonious Model Selection Based on Bayesian Information Criterion ........ 1
- Introduction ....................................................................................................................... 1
- Notation .......................................................................................................................... 2
- Theory ............................................................................................................................. 3
  - “Bag of Words” Model ............................................................................................. 3
  - PTM ............................................................................................................................. 4
  - Bayesian Information Criterion (BIC) Derivation ..................................................... 5
- Experiment ....................................................................................................................... 10
  - Generalized Expectation Maximization (EM) Algorithm .......................................... 10
  - Learn the Model ........................................................................................................ 11
  - Selecting the model order ......................................................................................... 13
- Result .............................................................................................................................. 13
  - Reuters -21578 ......................................................................................................... 14
  - 20-Newsgroups ......................................................................................................... 16
- Conclusion ...................................................................................................................... 17

Chapter 2  A Mixture-based Hybrid of DNN-based Transfer Learning and CNN-based
Supervised Learning for Infrared Image Object Recognition with Limited Labeled
Training Data ....................................................................................................................... 18
- Introduction .................................................................................................................... 18
- Transfer Learning ......................................................................................................... 18
- Caveats on transfer learning ....................................................................................... 20
- A Mixture of Transfer Learning Based DNN and CNN class posteriors .................... 22
- Theory ........................................................................................................................... 23
  - Standard Transfer Learning .................................................................................... 23
  - Hybrid Model ........................................................................................................... 24
  - Pre-trained Model on ImageNet ............................................................................... 25
  - Cifar-10 ..................................................................................................................... 25
- Result .............................................................................................................................. 26
  - Jointly Optimize all the Parameters ......................................................................... 27
- Conclusion ...................................................................................................................... 28

Reference ............................................................................................................................ 28

Appendix  Acknowledgements .......................................................................................... 30
LIST OF FIGURES

Figure 1-1: The comparison of our modified model and the PTM over BIC and hold out likelihood. .................................................................19

Figure 1-2: (a) The comparison of our modified model and the PTM over label consistency, (b) the ratio of words that use shared distribution under all topics. ..........20

Figure 1-3: The comparison of our modified model and the PTM over BIC and hold out likelihood. ........................................................................21

Figure 1-4: (a) The comparison of our modified model and the PTM over label consistency, (b) the ratio of words that use shared distribution under all topics. ........22

Figure 2-1: Examples of Cifar-10 data set of all 10 categories. .................................................................31

Figure 2-2: Comparison of three model over accuracy. .................................................................32

Figure 2-3: Comparison of three model over accuracy .................................................................33
Chapter 1

Parsimonious Model Selection Based on Bayesian Information Criterion

In this part we re-visited the paper ‘Parsimonious Topic Models with Salient Word Discovery [1]’. In the Parsimonious Topic Model (PTM) there are topic salient words which have topic-specific probabilities with the rest described by a universal model. A Bayesian Information Criterion (BIC) was derived to balance the goodness of fit and model complexity. Inspired by the work done in the ‘Unsupervised Learning of Parsimonious Mixtures on Large Spaces With Integrated Feature and Component Selection[2]’, in our work we changed the BIC by giving a different description of the cost of the structure of topic specific words; also we developed a new optimization method and got promising result from those changes.

Introduction

Topic modeling [3] is a type of statistical modeling for finding the ‘topics’ that occur in a collection of documents. The Latent Dirichlet Allocation (LDA) [4] topic model is one of the well-known topic models. The LDA topic model assume that each topic is a probability mass function defined over the given vocabulary, and for each of the documents, every word follows a document specific mixture over the topics. In the PTM paper, two shortcomings of LDA are discussed. Firstly, all the words have their own probability parameters under every topic; this strategy may use a huge amount of parameters and the model is prone to overfitting; secondly, in LDA every topic is assumed to present in every document, with a non-zero probability. However, PTM gives a more sparse description in the two aspects mentioned above. Firstly, some words are
not topic-specific, they use a universal shared model; secondly in each document only some of
the topics occur with a non-zero probability.

Bayesian Information Criterion (BIC) [5], [6] is a widely used criterion in model
selection; there are two parts in the negative logarithm of the Bayesian marginal likelihood: the
likelihood of the data and the model complexity cost. So we can use BIC to balance the fitting
goodness and the model complexity. The BIC proposed in PTM [1] improved the BIC in two
aspects. Firstly, the proposed BIC has differentiated cost terms based on different effective
sample size of different type of parameters; secondly, PTM introduced the shared feature
representation to decrease the feature dimension.

Our contribution is that we give a cheaper expression to describe the cost of topic specific
words and get an even more sparse model. Inspired by the intuition that in text corpora, there are
some words that are not related to any topics, we implemented an optimization method to jointly
optimize all the parameters related to a single word under each topics to reach the point that all
the word ‘switches’ related to a single word are closed. Our work improves the PTM model in
giving a more sparse and reasonable representation of the topic model.

Our model is to solves the unsupervised feature selection problem; firstly we assume the
hyper-parameter--number of topics--is known, and we get an optimized structure of the model by
determining the topic-specific words under each topic and which topics occur in each document.
Then we change the number of topics (hyper-parameter); for each value of the hyper-parameters
we compute the BIC, and optimized hyper-parameter is the one with the lowest BIC.

**Notation**

Suppose a corpus consists of $D$ documents and $N$ unique words, index $d \in \{1, 2, \ldots, D\}$
and $n \in \{1, 2, \ldots, N\}$ are document and word index respectively, and unique topic is indexed by
\( j \in \{1, 2, \ldots, M\}, \text{M is the total number of topics (model order). And here are some different definition in each document:} \)

\( L_d \) is the number of unique words in document \( d \).

\( w_{id} \in \{1, 2, \ldots, N\}, i = 1, \ldots, L_d \) is the \( i \)-th word in document \( d \).

\( v_{jd} \in \{0, 1\} \) is the topic ‘switch’, it indicates whether topic \( j \) is present in document \( d \).

Topic \( j \) is present in document \( d \) if \( v_{jd} = 1 \), otherwise \( v_{jd} = 0 \).

\( M_d \equiv \sum_{j=1}^{M} v_{jd} \in \{1, 2, \ldots, M\} \) is the number of topics that present in document \( d \).

\( \alpha_{jd} \) is the proportion for topic \( j \) in document \( d \).

In each topic:

\( \beta_{jn} \) is the probability that word \( n \) is topic specific under topic \( j \).

\( \beta_{0n} \) is the probability that word \( n \) follows a shared model.

\( u_{jn} \in \{0, 1\} \) indicates whether \( (u_{jn} = 1) \) or not \( (u_{jn} = 0) \) word \( n \) is topic-specific under topic \( j \).

\( N_{j} \equiv \sum_{n=1}^{N} u_{jn} \) is the total number of topic specific words under topic \( j \).

\( L_{j} \equiv \sum_{n=1}^{N} L_d v_{jd} \) is the sum of the length of the documents for which topic \( j \) is present.

\begin{itemize}
  \item \textbf{Theory}
\end{itemize}

\textbf{“Bag of Words” Model}

Bag of words model \cite{7} is commonly used in method of document classification where the count of each word is used as a feature for training a classifier. Using the Bag of Words model we can transform the text corpora into a feature matrix, where each row vector \( x = (x_1, x_2, \ldots x_D) \) in the matrix is a bag of each document, the length of the vector is the total number of the unique
PTM

We firstly introduce the data generation method of PTM.

1) For each document $d = 1, 2, \ldots, D$

2) For each word $i = 1, 2, \ldots, L_d$
   a) Randomly select a topic based on the probability mass function (pmf)
      \[ \{ \alpha_{jd}, j = 1, 2, \ldots, M \} \].
   b) Given the selected topic $j$, randomly generate the $i$-th word based on the topics
      pmf over the word space \( \{ \beta_{jn}^{u_{jn}}, \beta_{0n}^{1-u_{jn}}, n = 1, 2, \ldots, N \} \)

Based on the above data generation method, we can get the data likelihood of the corpus $D$ under our model \((H, \theta)\):

\[
p(D|H, \theta) = \prod_{d=1}^D \prod_{i=1}^{L_d} \sum_{j=1}^M [\alpha_{jd} v_{jd} \beta_{jn}^{u_{jn}} \beta_{0n}^{1-u_{jn}}]
\]

(1)

Where $v_{jd}$ is the topic switch that indicate whether topic $j$ is present in document $d$, it means that only if $v_{jd} = 1$, the related parameter $\alpha_{jd}, \beta_{jn}, \beta_{0n}$ are effective parameters. $u_{jn}$ is the word switch that indicates if the word $n$ is topic-specific under topic $j$. The model structure, denoted by $H\{v, u, M\}$, consists of two kinds of switches and the number of the topics $M$ (model order). Likewise, the model parameters are denoted by $\theta\{\{\alpha_j\}, \{\beta_{jn}\}, \{\beta_{0n}\}\}$. The model structure together with the model parameters constitute the PTM model.

In PTM the parameters are constrained by the following two conditions:
Firstly, we know that $\alpha_{jd}$ is the probability that topic $j$ is present in document $d$, and $v_{jd}$ determines whether topic $j$ is present. The probability must sum to one. So we have:

$$\sum_{j=1}^{M} \alpha_{jd} v_{jd} = 1, \forall d$$  \hspace{1cm} (2)

Also for the word parameters $\beta_{jn}$ and $\beta_{0n}$, we have:

$$\sum_{j=1}^{M} (u_{jn} \beta_{jn} + (1 - u_{jn}) \beta_{0n}) = 1, \forall j$$  \hspace{1cm} (3)

Based on the PTM described above, there are two aspects in the model training: training the model parameters $\Theta$ and selecting the model structure $H$. Assuming the model structure is known, we can estimate the model parameters using the Expectation Maximization (EM) algorithm, by introducing the hidden data, we compute the expected complete data log likelihood, and maximize it with the two constraints mentioned above. The model selection is more complicated, we need to derive a BIC to balance the model complexity and the data likelihood.

In the PTM model a Generalized Expectation Maximization (GEM) algorithm is proposed to update the model parameters ($\Theta$) and the model structure ($H$) iteratively. In the following section, we will introduce the derivation of BIC, the GEM algorithm proposed in PTM paper, and the data updating rule.

**Bayesian Information Criterion (BIC) Derivation**

In this part we will derive the BIC objective function, the naive BIC has the following form:

$$BIC = K \ln(n) - 2 \ln(\hat{L})$$  \hspace{1cm} (4)

Where $\hat{L}$ is the maximized value of the data likelihood of the model $H$, i.e., $\hat{L} = p(D|H, \hat{\Theta})$, where $\hat{\Theta}$ is the collection of parameters that maximize the likelihood function.

$n$ is the number of datapoints in the training sample.
K is the number of parameters estimated by the model.

However, the Laplace’s approximation used in deriving BIC is under the assumption that the feature space is really small compared with the sample size, but for our topic model, the feature space is large. Moreover, all the parameters penalize the BIC with the same sample size, but in the PTM different type of parameters contributes unequally to the model complexity. So a new BIC is derived.

So here we derive our BIC function, with which we jointly optimize the model structure H and the parameters θ. Given a certain model order M we update the parameters and two switches {v, u} iteratively to get a minimize BIC. Then the optimized M is chosen by selecting the M that gives the BIC-minimizing model.

The Bayesian approach of model selection is to maximize the posterior probability of the model H given the dataset D [6] , When applying the Bayes Theorem to calculate the posterior probability we can get:

\[ P(H|D) = \frac{P(D|H)P(H)}{P(D)} \]  

(5)

Here we define:

\[ I = P(D|H) = \int P(D|H, \theta)P(\theta|H)d\theta \]  

(6)

Where \( P(\theta|H) \) is the prior of the parameters given the model structure H. Then, we need to use the Laplace’s method to approximate I, given the knowledge that for large sample size \( P(D|H, \theta)P(\theta|H) \) is peaked around the maximum point (the posterior mode \( \hat{\theta} \)). We can rewrite I as:

\[ I = P(D|H) = \int \exp(\log(P(D|H, \theta)P(\theta|H)))d\theta \]  

(7)

We can now expand \( \log(P(D|H, \theta)P(\theta|H)) \) around the posterior mode \( \hat{\theta} \) using Taylor series expansion.
\[
\log(P(D|H, \theta)P(\theta|H)) \approx \log(P(D|H, \hat{\theta})P(\hat{\theta}|H)) + (\theta - \hat{\theta})\nabla_\theta Q|_{\theta} - \\
\frac{1}{2} (\theta - \hat{\theta})^T \Sigma_\theta (\theta - \hat{\theta})
\]  

(8)

Where \(Q \triangleq \log(P(D|H, \theta)P(\theta|H))\) and \(\Sigma_\theta = -\Sigma_\theta\), where \(\Sigma_\theta\) is the Hessian matrix.

\[
\Sigma_{i,j} = \partial^2 \frac{Q}{\partial \theta_i \partial \theta_j}|_{\theta = \hat{\theta}}
\]

Since \(Q\) attains its maximum at \(\hat{\theta}\), \(\nabla_\theta Q|_{\theta = \hat{\theta}} = 0\) and \(\tilde{\Sigma}_\theta = -\Sigma_\theta\) is negative definite. So we can approximate \(I\):

\[
I \approx P(D|H) = P(D|H, \hat{\theta})P(\hat{\theta}|H) \int e^{-\frac{1}{2}(\theta - \hat{\theta})^T \tilde{\Sigma}_\theta (\theta - \hat{\theta})} d\theta
\]

(9)

With the above form of the approximation, we can treat \(e^{-\frac{1}{2}(\theta - \hat{\theta})^T \tilde{\Sigma}_\theta (\theta - \hat{\theta})}\) as a scaled Gaussian distribution with the mean \(\hat{\theta}\) and covariance \(\tilde{\Sigma}_\theta\). So:

\[
\int e^{-\frac{1}{2}(\theta - \hat{\theta})^T \tilde{\Sigma}_\theta (\theta - \hat{\theta})} d\theta = (2\pi)^{k/2} |\tilde{\Sigma}_\theta|^{-1/2}
\]

(10)

\(k\) is the number of parameters in \(\theta\),

So we have the approximation of \(I\):

\[
I \approx P(D|H) = P(D|H, \hat{\theta})P(\hat{\theta}|H)(2\pi)^{k/2} |\tilde{\Sigma}_\theta|^{-1/2}
\]

(11)

The BIC is the negative log model posterior:

\[
BIC = -\log\left(\hat{I}P(H)\right) = -\frac{k}{2} \log(2\pi) + \frac{1}{2} \log(|\tilde{\Sigma}_\theta|) - \log\left(P(D|H, \hat{\theta})\right) - \\
\log\left(P(\hat{\theta}|H)\right) - \log(P(H))
\]

(12)

Note that \(P(\hat{\theta}|H)\) is the prior of the parameters given the structure \(H\), it can be treated as uniform distribution, thus a constant. So this term can be neglected. Now the \(\log\left(P(D|H, \hat{\theta})\right)\) term is the data likelihood, and \(k\) is the total number of all the model parameters. Now we need to compute \(\frac{1}{2} \log(|\tilde{\Sigma}_\theta|)\) and \(\log(P(H))\).
We know that \( \tilde{\Sigma}_a \) is the Hessian matrix of all the model parameters, based on the effective sample size idea proposed in [1], we can have an approximation of \( \log(|\tilde{\Sigma}_a|) \) by the number of parameters and the corresponding effective sample size.

\[
\frac{1}{2} \log(|\tilde{\Sigma}_a|) \approx \frac{1}{2} \sum_{d=1}^{D} (M_d - 1) \log(L_d) + \frac{1}{2} \sum_{j=1}^{M} \sum_{d=1}^{D} u_{jd} \log(L_j) + \frac{1}{2} \sum_{d=1}^{D} \log(\sum_{j=1}^{M} L_j) \tag{13}
\]

The terms at the right hand represent the cost of the model parameters \( \alpha, \beta, \beta_0 \) respectively, note that in naïve BIC form, each parameter pays \( \frac{1}{2} \log(\text{sample size}) \). Here we use the effective sample size, the effective sample size of \( \alpha_{jd} \) is \( L_d \), parameter \( \beta_{jn} \) has the sample size \( \bar{L}_j \) and the parameter \( \beta_{0n} \) has the sample size \( \sum_{j=1}^{M} \bar{L}_j \).

Another term to be estimated is \( \log(P(H)) = \log(p(v)) + \log(p(u)) \), for \( \log(p(v)) \), consider that in each document \( d \), suppose the number of topics follows a uniform distribution, and the switch configuration also follows a uniform distribution over all \( \binom{M}{M_d} \) configurations. We can estimate:

\[
-\log(p(v)) = D \log(M) + \sum_{d=1}^{D} \log(\binom{M}{M_d}) \tag{14}
\]

When estimate \( \log(p(u)) \) we proposed a model that can estimate \( \log(p(u)) \) and the corresponding parameters cost of \( \beta, \beta_0 \) jointly. For each word \( n \), there may be three types configuration of the word switches \( \{u_{jn}, j = 1, \ldots, M\} \), 1) each word is topic specific, i.e., \( \sum_{j=1}^{M} u_{jn} = M \); 2) all the words are not topic specific, i.e., \( \sum_{j=1}^{M} u_{jn} = 0 \); 3) some, but not all, components use the shared distribution, i.e., \( 0 < \sum_{j=1}^{M} u_{jn} < M \). For the case 1 and 2, there are only one possible configuration of the word switches related to a word (all open or all closed), so the probability is 1, for the case 3 there is a cost \( M \log(2) \) to specify which components use the
shared distribution from $2^M$ possible configurations. So we can estimate $-\log(p(u))$ together with those two terms
\[ -\log(p(u)) + \frac{1}{2} \sum_{d=1}^{\rho} \sum_{j=1}^{M_d} u_{jd} \log(\bar{L}_j) + \frac{1}{2} \sum_{d=1}^{\rho} \sum_{j=1}^{M_d} \log(\Sigma_{j=1}^{d} \bar{L}_j) \]

\[ = \sum_{n=1}^{N} \left( \frac{F_1(u_n)}{2} \log(\sum_{j=1}^{M} \bar{L}_j) + \frac{F_2(u_n)}{2} \sum_{j=1}^{M} \log(\bar{L}_j) + F_3(u_n) \left( \frac{1}{2} \log(\sum_{j=1}^{M} \bar{L}_j) + \frac{1}{2} \sum_{j=1}^{M} u_{jn} \log(\bar{L}_j) \right) \right) \] (15)

Where:

\[ F_1(u_n) = \begin{cases} 1, & \text{if } \sum_{j=1}^{M} u_{jn} = 0 \\ 0, & \text{otherwise} \end{cases} \]

\[ F_2(u_n) = \begin{cases} 1, & \text{if } \sum_{j=1}^{M} u_{jn} = M \\ 0, & \text{otherwise} \end{cases} \]

\[ F_3(u_n) = \begin{cases} 1, & \text{if } 0 < \sum_{j=1}^{M} u_{jn} < 1 \\ 0, & \text{otherwise} \end{cases} \]

Based on the estimation discussed above, we have our BIC:

\[ BIC = D \log M + \sum_{d=1}^{\rho} \log \left( \frac{M_d}{M_d - 1} \right) \log(\frac{L_d}{2\pi}) + \frac{1}{2} \sum_{d=1}^{\rho} (M_d - 1) \log(\frac{L_d}{2\pi}) \]

\[ + \sum_{n=1}^{N} \left( \frac{F_1(u_n)}{2} \log(\frac{\sum_{j=1}^{M} \bar{L}_j}{2\pi}) + \frac{F_2(u_{jn})}{2} \sum_{j=1}^{M} \log(\frac{\bar{L}_j}{2\pi}) + F_3(u_n) \left( \frac{1}{2} \log(\frac{\sum_{j=1}^{M} \bar{L}_j}{2\pi}) + \frac{1}{2} \sum_{j=1}^{M} u_{jn} \log(\frac{\bar{L}_j}{2\pi}) + M \log 2 \right) \right) \]

\[-\log(p(D | H, \Theta)) \]
Generalized Expectation Maximization (EM) Algorithm

The EM algorithm is a popular method in statistic estimation problems involving incomplete data. For some unsupervised learning task, we only have the data points $D$, but we don't have any label of those data points, so during the training process, we create the labels $Z$ which is called latent variables. The EM algorithm can be described as follow:

E-step: With the parameters fixed, we can compute the expectation of the latent variables $p(Z|D, \theta)$, which gives the class information of each data point, using the expectation of the latent variables we can compute the expectation of the incomplete data likelihood:

$$L_c = \sum_Z p(Z|D, \theta) \log(p(D, Z|\theta))$$  \hspace{1cm} (16)

M-step: update the parameters $\theta$ to find the maximum value of the complete data likelihood.

By doing the E-step and M-step iteratively, the complete data log-likelihood typically converges to a local optimum.

But note that in the PTM there are not only the model parameters $\theta$ but also the model structure $H$ we need to optimize over. The original EM algorithm doesn't work, however, a generalized expectation maximization (GEM) algorithm is proposed in some works related to topic models. The PTM paper also used a GEM algorithm to do the optimization.

Here we can also use GEM algorithm to optimize our BIC function. And our GEM algorithm is under the assumption that the model order $M$ are known. Firstly we introduce the hidden data $Z$, $Z_{id}$ is a $M$-dimensional binary vector, it represent which topic is the topic of origin for the word $w_{id}$. For example if the element $Z_{id}^{(j)} = 1$ and other elements of $Z_{id}$ are all equal to zero, we can know that topic $j$ is the topic of origin of the word $w_{id}$.
With the hidden data and our BIC function, our GEM algorithm are as follow:

In E-step, firstly we can compute expectation of the hidden data $Z$

$$P\left(Z_{id}^{(j)} = 1 \mid w_{id}, \theta^t, H\right) = \frac{P(w_{id} \mid Z_{id}^{(j)}; \theta^t, H) P(Z_{id}^{(j)} \mid \theta^t, H)}{\sum_{Z_{id}} P(w_{id} \mid Z_{id}^{(j)}; \theta^t, H) P(Z_{id}^{(j)} \mid \theta^t, H)}$$  (17)

We can treat $P(Z_{id}^{(j)} \mid \theta^t, H)$ as an uniform distribution. So:

$$P\left(Z_{id}^{(j)} = 1 \mid w_{id}, \theta^t, H\right) = \frac{\sum_{M} \sum_{\alpha_{id} v_{id}} \beta_{jn}^{u_{jw_{id}}} \beta_{on}^{1-u_{jw_{id}}}}{\sum_{i=1}^M \sum_{\alpha_{id} v_{id}} \beta_{jn}^{u_{jw_{id}}} \beta_{on}^{1-u_{jw_{id}}}}$$  (18)

In the generalized M-step, based on the expectation of the complete data BIC we computed in E-step, we update the model structure $H$ and the model parameters $\theta$ independently. Firstly we optimize the model parameters given fixed model structure. Then we optimize the model structure given fixed model parameters.

**Learn the Model**

When updating the model parameters, note that the only term in BIC that is related to the model parameters is the data likelihood term, so we can use the complete data likelihood computed in E-step to optimize the model parameters. Taking those two constraints into consideration, we have our objective function.

$$L = \sum_{d=1}^D \sum_{i=1}^{L_d} \left[ v_{jd} P\left(Z_{id}^{(j)} = 1 \mid w_{id}, \theta^t, H\right) \left( \log(\alpha_{jd}) + u_{jw_{id}} \log(\beta_{jw_{id}}) + (1 - u_{jw_{id}}) \log(\beta_{0w_{id}}) \right) - \sum_{d=1}^D \lambda_d \left( \sum_{i=1}^n \alpha_{jd} v_{jd} - 1 \right) - \sum_{j=1}^M \mu_j \sum_{i=1}^n \left( u_{jn} \beta_{jn} + (1 - u_{jn}) \beta_{on} \right) - 1 \right]$$  (19)

By computing the partial derivatives of different parameters and setting those derivatives to zero we can get the optimized model parameters as follow.
\[ \alpha_{jd} = \frac{\sum_{i=1}^{l_d} p(Z_{id}^{(j)} = 1 | w_{id}; \theta^t, H) v_{jd}}{\sum_{i=1}^{M} \sum_{i=1}^{l_d} p(Z_{id}^{(j)} = 1 | w_{id}; \theta^t, H) v_{id}} \]  

(20)

\[ \beta_{jn} = \frac{u_{jn} \sum_{d=1}^{D} \sum_{i=1}^{l_d} p(Z_{id}^{(j)} = 1 | w_{id}; \theta^t, H) v_{jd}}{\mu_j} \]  

(21)

Where \( \mu_j \) is the Lagrange multiplier, we can compute it by multiplying both side of (21) by \( u_{jn} \), summing over all \( n \), and applying the distribution constrain on topic \( j \):

\[ \mu_j = \frac{\sum_{n=1}^{N} u_{jn} \sum_{d=1}^{D} \sum_{i=1}^{l_d} p(Z_{id}^{(j)} = 1 | w_{id}; \theta^t, H) v_{jd}}{1 - \sum_{j=1}^{M} (1 - u_{jn}) \beta_{0n}} \]  

(22)

For the shared parameters, we didn’t use this method to estimate, however we only estimate them once via the global frequency counts at initialization and hold them fixed during the GEM algorithm.

\[ \beta_{0n} = \frac{\sum_{d=1}^{D} \sum_{i=1}^{l_d} w_{id} = n}{\sum_{d=1}^{D} l_d} \]  

(23)

When updating the model structures, we implemented iterative loop in which all the switches are visited one by one, if the current change reduce the BIC, we accept the change, otherwise we keep the switch unchanged. But here note that in updating the word switches, we updating all the word switches of a single word jointly to see if it is possible go get the point that all the switches related to a single word are closed. This process is repeated over all the switches until there is no decrease on BIC or reach the pre-defined max number of loop. We update the topic switches \( v \) and the word switches \( u \) independently, firstly we update the word switches \( u \) until convergence, then we update the topic switches \( v \) until convergence. Then we back to E-step.

Note that when updating the word switches, for each single search of all the switches related to one word, we have three configurations. All switches are closed, all switches are open,
and some are closed some are open, we compute the minimized BIC of each configuration, and then choose the configuration that has the lowest BIC value.

Selecting the model order

The optimization process discussed above is under the assumption that the model order is known. The model order selection is based on doing the optimization process under different model orders. We covered different model orders with a top-down fashion. We initialize the model with a specific number of topics and reduce the number by a predefined step size, for the model trained in each step, we remove the topics with the smallest aggregate mass. This process is applied iteratively until the predefined minimum order is reached.

Result

We trained our model on two famous dataset: Reuters-21578 and 20-Newsgroup corpora. For each of those dataset, we compared our model with the PTM, with respect to the held-out likelihood, the label purity, and BIC.

When computing the hold-out likelihood we divide the documents in the test set into two parts, observed part and the hold-out part. Firstly we compute the topic proportions based on the observed part then compute the hold-out log-likelihood based on the hold-out part.

\[
\sum_{d=1}^{D} \sum_{i=1}^{L_d} \log(\sum_{j=1}^{M} E_q[\alpha_{jd}] E_q[\beta_{ij/d}])
\]

(24)

In our model \(E_q[\alpha_{jd}]\) is directly the topic proportions \(\alpha_{jd}\), \(E_q[\beta_{ij/d}]\) is \(u_{jn}\beta_{jn} + (1 - u_{jn})\beta_{0n}\).
In our experiment, we use some labeled text documents, this makes it possible for us to use label purity to evaluate our model. So we also compare the label purity of our model and the PTM.

**Figure 1-1** The comparison of our modified model and the PTM over BIC and hold out likelihood.

**Reuters -21578**

Reuters-21578 is a widely used test collection for text categorization research, there are in total 7674 documents from 35 categories. For this dataset we firstly process this dataset with stemming and stop word removal. After processing there are 17387 unique words.
For both PTM and our modified model, we do the same experiment. We initialize the model with 100 topics and delete 2 topics in each step. We record the BIC and hold out likelihood of both model and report it in Fig. 1-1. We can see that our modified model has higher hold out likelihood at almost all the numbers of topics, and lower BIC when the number of topics is larger than 36. In Fig. 1-2 we compare the label purity to see if our model really did a better job, also the ratio of the word use shared distribution under all topics is reported to show the sparsity of our model. It shows that our model has higher label purity.

Figure 1-2 (a) The comparison of our modified model and the PTM over label consistency, (b) the ratio of words that use shared distribution under all topics.

For both PTM and our modified model, we do the same experiment. We initialize the model with 100 topics and delete 2 topics in each step. We record the BIC and hold out likelihood of both model and report it in Fig. 1-1. We can see that our modified model has higher hold out likelihood at almost all the numbers of topics, and lower BIC when the number of topics is larger than 36. In Fig. 1-2 we compare the label purity to see if our model really did a better job, also the ratio of the word use shared distribution under all topics is reported to show the sparsity of our model. It shows that our model has higher label purity.
20-Newsgroups

20-Newsgroups is another dataset we used to compare our modified model with the PTM. There are 11269 documents from 20 classes. There are 53976 words after stemming and stop word removal. We initialize the model with 100 topics and delete 2 topics in each step. Here is the result.

Figure 1-3 The comparison of our modified model and the PTM over BIC and hold out likelihood
Conclusion

In this part of work we revisited the PTM model, give a cheaper representation of the BIC, and implement a different method to update the word switches. The result shows that our modified model can give us a more sparse model, with higher held out likelihood and lower BIC.
Chapter 2

A Mixture-based Hybrid of DNN-based Transfer Learning and CNN-based Supervised Learning for Infrared Image Object Recognition with Limited Labeled Training Data

In this part of work, we implemented a model which combined the Transfer Learning and the Convolutional Neural Network (CNN). We make use of the pre-trained Deep Neural Network (DNN), which can help us to extract the informative features from an image. Also we train our own CNN to do the image classification. We compared the Transfer Learning model and CNN under different data size, then we found that a hybrid model which combine both the Transfer Learning model and CNN get a better performance in some range of data size.

Introduction

Transfer Learning

One of the most practically valuable machine learning paradigms proposed in recent years is that of transfer learning [8], [9], particularly for classification and recognition on image domains. Here, one has a niche classification/recognition domain of interest, e.g. recognition of image scenes with military vehicles and discrimination from scenes either with civilian vehicles or with no vehicles. For a particular image modality, one may possess only a (relatively) small number of labeled image examples from the given categories (e.g., ‘military’ and ‘non-military’) of interest. While in principle a deep neural network (DNN) (or even just a relatively “shallower” convolutional neural network (CNN) [10]) should be a very suitable, effective classifier model for accurately discriminating these categories, the number of (NN weight) free parameters that may
need to be learned to build this (highly effective) model could be in the tens to hundreds of
thousands. Directly training a model of this size given only e.g. hundreds of labeled training
images will lead to severe overfitting, resulting in a model with poor generalization accuracy. On
the other hand, one can try to avoid overfitting by matching the complexity of the model to the
size of the available labeled training set. This can be accomplished e.g. by use of fixed
(untrained) image descriptors as features (e.g. SIFT, visual bag of words, or other common image
descriptors), rather than by learning the key image features through learning the weights of the
many layers of a DNN. One can then feed this fixed feature vector representation of the image
into a more “parameter-light” classifier model – either a shallow neural network or e.g. a support
vector machine (SVM). The problem with this approach is that this fixed (unlearned) feature
representation may fail to capture the key discriminating features for the classification domain of
interest – i.e., the learned classifier, while not overfitting, may still exhibit high model bias, with
the result again poor (generalizable) discrimination power of the model.

Transfer learning captures the best attributes of both of these learning approaches. On the
one hand, the features are learned from image data, so they tend to be better suited for class
discrimination than use of fixed feature representations. On the other hand, the parameters that
need to be learned based on the (limited) niche labeled training set are kept to a modest number.
Thus, the learned model should not manifest severe overfitting problems. All of these are
achieved by leveraging a massive labeled training repository of images (e.g., ImageNet) which,
while in general not containing any images with objects that well-match categories of interest for
the niche domain, still does contain images possessing key features of images from these niche
categories (e.g., edges at particular orientations, particular textures, etc.). The ImageNet domain,
for example, consists of one million labeled images from 1000 defined categories – a DNN
classifier has already been built on this 1000-category domain and is publicly available. The
presumption of transfer learning is that the feature extraction performed by this DNN classifier is
essentially universal, extracting the key features useful for discriminating images for almost any niche image classification domain (with categories that may have absolutely nothing to do with the ImageNet categories). Thus, transfer learning to the niche domain is performed as follows:

1. Feed the niche application training images into the pre-trained (e.g., ImageNet) DNN, performing image resampling as needed to ensure the niche images are of the dimensionality expected as input by the DNN.

2. Extract as a feature vector for each such image the output of the penultimate layer of the DNN (the layer just before the final, decision layer). This feature vector (call it $z_i$ for the $i$-th niche training image), which contains the crucial features used for discriminating between the 1000 Imagenet categories, typically consists of a few thousand features.

3. Treat this feature vector as the input to a simple classifier (a shallow neural net, a logistic function, or a support vector machine) and train the parameters of this simple classifier model in a supervised fashion, using the training pairs $\{(z_i, c_i), i = 1, \ldots, N_{\text{niche}}\}$, where $c_i$ is the class label for the $i$-th training image and where $N_{\text{niche}}$ is the number of niche labeled training images.

4. Testing: Feed a test image, $x$, into the pre-trained DNN, extract the penultimate layer feature vector $z$, and feed this as input to the niche classifier to make a decision.

In a variety of application domains, this approach has been found to yield superior classification results to those obtained by conventionally training a classifier (using standard, fixed feature representations) directly on the niche domain.

**Caveats on transfer learning**

First, some works simply report (presumably) good classification accuracy obtained on a given domain using a transfer learning approach without benchmark-comparing the obtained
accuracy against that of a conventional (even CNN) classifier trained directly on the labeled data available for the niche domain, e.g. [11]. Such works have not established that transfer learning is crucial for their domain – the pre-trained DNN uses convolutional neural network (CNN) feature extraction layers – it could simply be that this CNN-based feature extraction, whether it is learned on the “universal” ImageNet domain or on the niche image domain’s training set, is the crucial factor leading to good accuracy. This hypothesis is not tested, e.g. in [12], and also in many other papers. All such papers are simply demonstrative that transfer learning works “pretty well” – not that it is the best approach to take for a given domain. The implication that we draw for research in this area is the following: it could be either that i) some categories or ii) even some sub-categories (that may not even be explicitly defined and labeled as such), for which there is “sufficient” niche training data, may be better modelled by a customized CNN (learned on the niche domain’s training set), rather than via the “universal” DNN’s feature representation coupled with transfer learning. How can this possibility be practically exploited, via a suitably designed classification system? One approach is this: i) train a transfer learning based classifier for the niche domain; ii) train a conventional CNN for the niche domain; iii) train a third classifier whose job is to predict which of these classifiers will make a more reliable decision on a given (test) image. While such an approach is possible, one of its limitations is that it forces making hard decisions, for a given test image, on which classifier to use. Making hard decisions in general entails information loss, which may compromise accuracy. We thus next instead propose a statistical modelling (mixture modelling) approach to best exploit both transfer learning and conventional supervised learning approaches in yielding improved niche domain classification. Our approach will jointly learn a transfer learning posterior model, a conventional (CNN) posterior model, and a (soft) classifier that determines the relative weight given to each of these two posteriors, in making posterior inferences on a given (test) image.
A Mixture of Transfer Learning Based DNN and CNN class posteriors

Let \( \mathbf{z} = g^{(L-1)}(x) \) be the universal DNN-derived feature vector for a DNN with L layers when the input to the DNN is the image \( x \). We propose to form a DNN feature vector dependent mixture of class posteriors:

\[
P[C = c|x] = P[M = 'TL'|z]P_{DNN}[C = c|z; \Gamma^{(L)}] + P[M = 'CL'|z]P_{CNN}[C = c|x; \Theta]
\]

Here, \( P[M = 'TL'|z] + P[M = 'CL'|z] = 1 \) with \( P[M = 'TL'|z] \) and \( P[M = 'CL'|z] \) the mixture component priors (mass) on the transfer learning (TL) and conventional CNN learning (CL) posteriors, respectively. We emphasize that we make this mixture component probability mass function DNN feature vector-dependent – in particular, we choose the logistic function:

\[
P[M = 'TL'|z] = \frac{e^{\beta^T z}}{1 + e^{\beta^T z}}
\]

That is, the contribution to the overall posterior \( P[C = c|x] \) from the two component posteriors is image-dependent. This flexibility in the mixture allows a given \( x \) to be classified primarily using either the transfer learning based posterior, using the conventional CNN posterior, or via some combination lying between these two extremes. Certain image types (categories or, again, sub-types that are not even explicitly defined) that are well-represented in the niche training set may (if the model has been suitably learned) assign a larger mixing weight to ‘CL’ than to ‘TL’. Likewise, image types under-represented in the niche training set may be better represented by the ‘TL’ component. Note also that we make the mixture component masses a function of \( z \), not \( x \). This limits the number of model parameters (the dimension of \( \beta \)) needed to represent the mixture component masses. The above is really a type of mixture of experts model, e.g. [?].
The parameters of this mixture model that need to be learned, based on the niche labeled training set, are: i) the mixture component pmf’s weight vector $\beta$; ii) The parameters $\Gamma^{(L)}$ specifying the decision layer of the DNN’s posterior (all internal DNN parameters are of course fixed to the pre-trained DNN parameter values); iii) The parameters specifying the conventional CNN’s posterior, $\Theta$ – this consists of all of the weights of the CNN (which could e.g. have five or more layers, including an input layer, a few convolutional layers, max-pooling layers, and an output layer). The model should be trained to maximize the overall posterior’s log-likelihood on the niche training set. Such training is also mathematically equivalent to a Kullback divergence (or cross entropy) minimization, where the “target” distributions are binary distributions – zero probability for all classes excepting the true class, which has probability one. Model learning is thus based on a (mixture-customized) variant of back propagation, with either a true gradient descent procedure or related mini-batch or stochastic gradient descent invoked. The hyper-parameters of the model and its learning, including the number of layers of the CNN and their sizes, the learning rate for gradient descent (or parameters that specify how this learning rate changes with iterations), and the learning stopping condition can be chosen in a standard way, e.g. via cross validation.

**Theory**

**Standard Transfer Learning**

For standard transfer learning, the class posterior model for the target domain is:

$$P_{DNN}[C = k | z; \Gamma^{(L)}] = \frac{e^{w_kz+b_k}}{\sum_{j=1}^{M} e^{w_jz+b_j}}$$

(2-1)
where the vector $z$ is obtained from the penultimate layer of the deep neural network, trained on a huge training set repository, and where the weights $\{w\}$ and bias $\{b\}$ are the parameters of the posterior model for the target domain. The training objective function for leaning the target domain’s parameters is the cross entropy (equivalently, the negative posterior log-likelihood):

$$J = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{M} y_i \log(P_{DNN}[C = j|z_i; \Gamma^{(L)}])$$

(2-2)

Where $M$ is the number of classes for the target domain and $N$ is the number of labeled data points for the target domain (which is much smaller than the number of labeled data points in the huge repository, which is also the reason we need transfer learning). $(z_i, y_i)$ is the $i$-th data point.

We can compute the gradients for gradient-based minimization of $J$ as follows:

$$\frac{\partial J}{\partial w} = \frac{1}{N} Z^T (P_t - 1_{y=1})$$

(2-3)

$$\frac{\partial J}{\partial b} = \frac{1}{N} 1_{1 \times N} (P_t - 1_{y=1})$$

(2-4)

Where $P_t$ is a $N \times M$ matrix, where $P_t(i, j) = P_{DNN}[C = j|z_i; \Gamma^{(L)}]$, $1_{y=1}$ is the one hot label, and $Z = (z_1^T, z_2^T, ... z_N^T)^T$

**Hybrid Model**

We assume that we already have a CNN trained in a standard way for the target domain, and get the probability matrix $P_{CNN}$ from it. Also from the transfer learning model we can also get the probability matrix $P_t$ and the extracted feature vector $z$. Again our objective function is

$$J_{hy} = \sum_{i=1}^{N} \sum_{j=1}^{M} y_i P[C = j|x_i]$$

(2-5)

So the gradients of the hybrid model’s parameters are:
\[
\frac{\partial J}{\partial \beta} = \frac{1}{N} Z^T \ast (1 / P(P_t - P_{CNN}) \ast P[M = 'TL'[z] \ast P[M = 'CL'[z])
\]

(2-6)

**Pre-trained Model on ImageNet**

ImageNet is a database of over 15 million hand-labeled, high-resolution images in roughly 22,000 categories. This database is organized according to the WordNet. In WordNet each concept is called a synset, each synset is a node in the ImageNet, each node has more than 500 images.

When talking about the ImageNet dataset, people mostly refer to the dataset used in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) rather the whole ImageNet dataset. The ILSVRC was founded in 2010 to improve the state-of-the-art technology for object detection and image classification on a large scale. There are many DNN models proposed in ILSVRC that have good performance in ImageNet image classification. And the pre-trained model are available online.

In our work, we used the ResNet-V2-101 model, which is 110 layers deep, trained on the ImageNet dataset and has 77.0% top-1 accuracy and 93.7% top-5 accuracy. The feature vector \( z \) is extracted from the ‘logits’ layer, which is the second to last layer of the ResNet DNN and gives a 1001 dimensional feature vector.

**Cifar-10**

We used the CIFAR-10 dataset as the target domain.
The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dataset is divided into five training mini-batches and one test batch, each with 10000 images.

For the CNN and Transfer Learning model, we use the data in the CIFAR training set to train the CNN and Transfer Learning model (we change the number of training batches used in different experiments from 1 to 5, with 10000 images in each batch). Then we used the same data used to train the CNN and Transfer Learning model in each experiment to train our hybrid model; the test set data is used to test all three models.

Figure 2-1 Examples of Cifar-10 data set of all 10 categories. (From the official website of cifar-10: https://www.cs.toronto.edu/~kriz/cifar.html)
Note that the accuracy of the CNN model increases significantly when trained with more data, while there is only slight increase for the transfer learning model. When the accuracies of the CNN model and standard transfer learning model are close, the hybrid model does a better job (improves the test set accuracy). Thus, we have preliminarily validated that hybrid transfer learning can outperform standard transfer learning, on a widely used target image domain (CIFAR-10).

**Jointly Optimize all the Parameters**

In this part we introduce another optimization method, that is, for the whole model, we jointly optimize all the parameters including the CNN weights, the weights in the Transfer Learning decision layer, and the weights in computing the model posterior. It is hard to compute the gradient manually, luckily we can make use of some built-in function in tensorflow to do this work.

Here are the results when we use this optimization method.
Conclusion

We compared the performance of the Transfer Learning and Convolutional Neural Network under different sample size. As the experiment shows, the transfer learning gives better accuracy when the sample size is really small, because it is a simple model with few parameters. With the increasing of the sample size, CNN becomes better. Then we proposed a mixture-based hybrid model that combine the idea of Transfer Learning and CNN, which gives accuracy better than both TF and CNN in a medium sample size.

Reference

Appendix

Acknowledgements

This work is supported by NGA STTR Phase 1, Toyon Research Corporation. I highly appreciate Dr. David Miller’s effort on guiding me the thesis. Thanks to Dr. Constantino Lagoa and Dr. Kultegin Aydin for reviewing my thesis.