A Model for Named Data Networking Caching Policies Inspired by Nonlinear Dynamical Systems

António Rodrigues (up200400437@fe.up.pt)

***Abstract*—At the time of its inception, the Internet mostly served the purposes of communication between connected end-hosts. Now, at the World Wide Web era, the Internet is immersed in a content-centric paradigm, more concerned about content generation, sharing and access. Recently, a new research trend — Information Centric Networking (ICN) — started advocating for deep modifications on the Internet's network layer, making it content-centric by design, including the widespread use of in-network caching.**

In this paper, we focus on the analysis of cache behavior in a specific ICN architecture — Named Data Networking (NDN) — under different caching policies, network topologies and content usage characteristics. To do so, we specify a simple and but modular NDN router model, loosely inspired in nonlinear dynamical systems. We implement the specified model in MATLAB, providing some simulation results with X simple caching policies, specifically (...).

I. Introduction

Departing from its initial model as a network for host-to-host communications, the Internet started shifting towards a content-centric model with the advent of the World Wide Web in the 1980s. This model persisted, and with increasingly demanding usage requirements, leading to the development of technologies such as Content Delivery Networks (CDNs) and Peer-to-Peer (P2P) networks [1]. These were built around the architecture's edge, due to the so-called 'ossification' [2] of the Internet's core, leading to inefficiencies in terms of latency, bandwidth usage, among others. Given the widespread adoption of the content-centric model, researchers to think about new and clean-slate designs for the Internet's core, in order for it to natively cope with these issues. Among such efforts [3], the research field of Information Centric Networking (ICN) [4] emerged, advocating the deliberate abolition of network locators, replacing of IP addresses with content identifiers and calling for the widespread use of in-network caching, so that content can be easily served from multiple anywhere in the network [5]–[10]. Here we focus on the aspect of in-network caching in one of such clean slate designs, the Named Data Networking (NDN) architecture [6].

In this paper, we focus on the analysis of cache behavior in NDN networks under different caching policies, network topologies and content usage characteristics. To do so, we specify a simple and but modular NDN router model, loosely inspired in nonlinear dynamical systems [11]. We implement the specified model in MATLAB, providing some simulation results with X simple caching policies, specifically (...).

The remainder of this paper is organized as follows. In Section II we provide an overview over the NDN architecture, focusing on the basic operation of its forwarding engine and the way it involves in-network caching. In Section III, we present the overall methodology followed during this work, including an explanation of the considered NDN router models, caching policies and network topologies, while in Section IV we show details about the implementation of such models in MATLAB. In Section V we present a set of experiments ran over our model implementation, as well as the respective results. Finally, in Section VI we draw some pertinent conclusions from the presented work.

II. Named Data Networking (NDN)

In the Named Data Networking (NDN) [6] architecture, clients issue subscriptions for content objects by specifying a hierarchical (URL-like) content name, e.g. `/pdeec/mtsp/2014/`, which is directly used in NDN packets. Destination network locators (e.g. IP addresses) are not used in this case, as NDN routers are able to forward such packets towards appropriate content-holding destinations, solely based on such names. NDN contemplates two fundamental types of packets, 'Interest' and 'Data' packets, used for content subscriptions and publications, respectively. Interest packets are originally released into the network by clients willing to access a particular content, addressing it via its content name, while Data packets carry the content itself.

An NDN router is conceptually composed by three main elements: (1) a Forward Information Base (FIB), (2) a Pending Interest Table (PIT) and (3) a Content Store (CS) [6]:

- **Forward Information Base (FIB):** Routing/forwarding table holding entries which relate a name prefix and a list of router interfaces to which Interest packets matching that content name prefix should be forwarded to.
- **Pending Interest Table (PIT):** A table which keeps track of the mapping between arriving Interest packets and the interfaces these have been received from, in order to save a reverse path for Data packets towards one or more subscribers (this may be a 1:N mapping, as an Interest packet matching the same content may be received in multiple interfaces).
- **Content Store (CS):** A cache for content, indexed by content name or item. This novel element allows for content storage at the network level. In-network caching allows an Interest to be satisfied by a matching Data packet in any location other than the original producer of the content, constituting one of the main content-oriented characteristics of NDN.

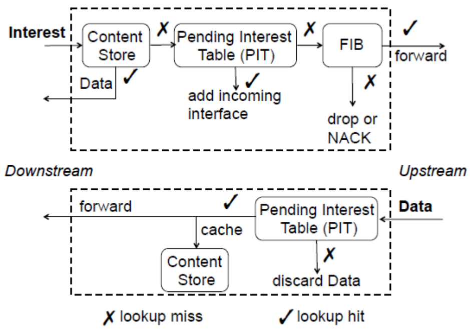

Fig. 1: Interest and Data packet processing according to NDN's forwarding engine [12].

In NDN, communication is receiver-driven, i.e. having the desire to fetch a particular content, a client releases an Interest packet into the network so that it is forwarded towards an appropriate content holder. In Figure 1 [12], we provide a graphical description of the mechanics of the forwarding engine of an NDN router, supported by the textual description provided below:

1) An Interest packet arrives on an interface (e.g. `iface0`) of an NDN router.
2) A longest prefix match on the content name specified in the Interest (e.g. `name`) is performed. The NDN router will now look in its CS, PIT and FIB, in that order, in order to resume the forwarding action:
    a) If there's a match in the router's CS, a copy of the respective CS entry will be sent back via `iface0`, the Interest packet is dropped. Depending on the pre-specified caching policy (e.g. MRU, LRU, LFU [1], etc.), the organization of the CS may change at this point. **End.**
    b) Else if there is an (exact) match in the PIT, `iface0` is added to the mapping list on the respective entry. The Interest packet is dropped (as a previous one has already been sent upstream). **End.**
    c) Else if only a matching FIB entry is found, the Interest packet is forwarded upstream, via all remaining interfaces on the list (except `iface0`), towards an eventual content holder. A PIT entry <`name`, `iface0`> is added. **End.**
    d) Else if there is no match at all, the Interest packet is simply discarded. **End.**

Note that in NDN only Interest packets are forwarded: intermediate NDN routers (i.e. between client and content holder) forward the Interests and have their respective PIT tables updated with Interest-to-interface mappings, pre-establishing a reverse path for Data packets to follow as soon as a content holder is found. When the reverse path is 'followed' (i.e. in the 'downstream' direction, lower part of Figure 1), each intermediate NDN router receiving a Data packet looks in its PIT for <`name`, `iface`> entries, and forwards the Data packet through all matching interfaces. In addition, a CS entry is created to cache the content locally at the router (again, depending on the caching policy, the organization of the CS may change at this point). If a Data packet with no matching PIT entries arrives, it is treated as unsolicited and discarded.

[1] http://en.wikipedia.org/wiki/Cache_algorithms.

III. Methodology

The work presented here follows the consulted background literature about supervised and semi-supervised text classification [13]–[15], reflecting our understanding of the material.

A. Notation

Text classification mainly deals with categorizing a set of text articles into some topic, according to a set of features such as the words contained in them. Here we present the notation to formalize these concepts.

An article a, belonging to a topic (i.e. a label/class) t, is represented by an array $a = \{w_1, w_2, ..., w_{|\mathcal{D}|}, t\}$, i.e. a list of $|\mathcal{D}|$ features and its label. In this case, each feature typically corresponds to the number of occurrences of the word w_i, the i-th word in a dictionary $\mathcal{D}$, within an article a_i. Other options for representing word frequency exist, such as Term Frequency and Inverse Document Frequency (TF-IDF) [15].

As an example, consider an article set composed by two instances, $\mathcal{A} = \{a_1, a_2\}$, each one of them with the following contents:

$$a_1 : \texttt{You like potato, I like potato.}$$

$$a_2 : \texttt{I say tomato, you say tomato.}$$

In this case, the dictionary $\mathcal{D}$ would contain the following words [2]:

$$\mathcal{D} = \{\texttt{You}, \texttt{like}, \texttt{potato}, \texttt{I}, \texttt{say}, \texttt{tomato}\}$$

The articles a_1 and a_2 could then be codified (using the sparse 'bag-of-words' approach [16]) as:

$$a_1 = \{1, 2, 2, 1, 0, 0\}$$

$$a_2 = \{1, 0, 0, 1, 2, 2\}$$

B. Generative Models for Text Classification

McCallum et al. [13] present two generative models for text classification, (1) a Multivariate Bernoulli event model and (2) a Multinomial event model: the first case considers multivariate Bernoulli as the parametric distributions describing each mixture component, only capturing the (non-) occurrence of word events in articles, while the second case considers Multinomial distributions, now capturing the quantity of word events. The authors state that the multinomial event model generally outperforms the multivariate Bernoulli model, specially when considering large dictionary sizes [13]. For the remainder of this paper, we focus our attention on the Multinomial event model.

[2] As we will see in Section V-A, in practice, the words in $\mathcal{D}$ might be converted to a common format, e.g. all words converted to lowercase, stripped of punctuation, etc.

Despite our focus on the Multinomial model, being GMs, both models assume (1) that an article a is generated according to a mixture model, encompassing several mixture components $c_j \in \mathcal{C} = \{c_1, c_2, ..., c_{|\mathcal{C}|}\}$. The shape of each component distribution is governed by a set of parameters θ. It is also assumed (2) there is a 1:1 correspondence between the components in $\mathcal{C}$ and topics of articles t_i.

One can look at the process of 'generating' an article a_i in the following manner: (1) Selecting a component c_j from the mixture model, with probability $P(c_j|\theta)$; (2) letting c_j generate a_i according to its own distribution $P(a_i|c_j, \theta)$. This results in the probability of an article a_i being generated by a component c_j:

$$P(c_j|\theta)P(a_i|c_j, \theta) \tag{1}$$

As different components in $\mathcal{C}$ can contribute to origin a similar article a_i, the probability of finding an article a_i is obtained by marginalizing expression 1 over all the components in $\mathcal{C}$:

$$P(a_i|\theta) = \sum_{j=1}^{|\mathcal{C}|} P(c_j|\theta)P(a_i|c_j, \theta) \tag{2}$$

C. Multinomial Naive Bayes

The Multinomial Naive Bayes (MNB) model described by McCallum et al. in [13] captures the frequency of words in articles and applies a 'bag-of-words' approach for article representation.

The second step in the process of generating an article a_i — related to the $P(a_i|c_j, \theta)$ term in expression 1 — can be further expressed as a sequence of $|a_i|$ draws (with replacement) of words w_k from the dictionary $\mathcal{D}$. Besides the assumptions mentioned in Section III-B, the MNB model considers that (1) the length of an article $|a_i|$ (word count) is independent of the topic/mixture model component c_j; and (2) the appearances of words in an article are conditionally independent from each other, given an article topic: the so-called 'Naive Bayes' assumption. The class conditional probability of an article a_i can then be thought of as a multinomial distribution over words, with a_i independent trials, in the form:

$$\begin{aligned} P(a_i|c_j, \theta) &= |a_i|! \prod_{k=1}^{|\mathcal{D}|} \frac{P(w_k|c_j, \theta)^{N_{i,k}}}{N_{i,k}!} \\ &\propto \prod_{k=1}^{|\mathcal{D}|} P(w_k|c_j, \theta)^{N_{i,k}} \end{aligned} \tag{3}$$

where $N_{i,k}$ is the number of times word w_k of a dictionary $\mathcal{D}$ appears in an article a_i. In practice, as the multinomial coefficient $\frac{|a_i|!}{N_{i,1}!...N_{i,|\mathcal{D}|}!}$ does not depend on the mixture components c_j, it is often ignored when the purpose is to maximize the likelihood $P(a_i|c_j, \theta)$ [14], [16], [17].

The set of mixture model parameters θ to be estimated during the training phase of the classifier consists of (1) each of the class conditional probabilities of words $\hat{\theta}_{w_k|c_j} \equiv P(w_k|c_j, \hat{\theta})$; and (2) the prior probabilities for each topic/mixture model component $\hat{\theta}_{c_j} \equiv P(c_j|\hat{\theta})$.

These estimates are obtained by Maximum a Posteriori (MAP) estimation of the parameters, i.e. we find the values of θ that maximize the posterior probability $P(\theta|\mathcal{A})$:

$$P(\theta|\mathcal{A}) \propto P(\theta) \times P(\mathcal{A}|\theta) \tag{4}$$

Here we follow the same representation of $P(\theta)$ used by Nigam et al. [14], a Dirichlet distribution, with $\sigma = 2$. The probability $P(\mathcal{A}|\theta)$ is equal to $\prod_{i=1}^{|\mathcal{A}|} P(a_i|\theta)$, as one assumes the article generation events are independent between each other, given the mixture model parameters θ.

The results of the MAP estimation reduce to 'counting problems'. Specifically, $\hat{\theta}_{w_k|c_j}$ is given by the ratio of appearances of a word w_k within all articles a_i belonging to a component c_j vs. the total number of word events for c_j:

$$\begin{aligned} \hat{\theta}_{w_k|c_j} &\equiv P(w_k|c_j, \hat{\theta}) \\ &= \frac{\alpha + \sum_{i=1}^{|\mathcal{A}|} N_{i,k} P(t_i = c_j|a_i)}{\alpha|\mathcal{D}| + \sum_{s=1}^{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{A}|} N_{i,s} P(t_i = c_j|a_i)} \end{aligned} \tag{5}$$

where $P(c_j|a_i) \in \{0, 1\}$ depending on the label of an article a_i and $N_{i,k}$ is the number of occurrences of the word w_k in an article a_i. Notice the inclusion of 'smoothing priors' α, used to avoid probabilities equal to zero in the lack of particular word events for a component c_j. In [13], [14] the authors use $\alpha = 1$, which is designated by Laplace smoothing.

The parameters $\hat{\theta}_{c_j}$ are given by the ratio of the articles belonging to a component c_j vs. the total number of articles $\mathcal{A}$:

$$\hat{\theta}_{c_j} \equiv P(c_j|\hat{\theta}) = \frac{\alpha + \sum_{i=1}^{|\mathcal{A}|} P(t_i = c_j|a_i)}{\alpha|\mathcal{C}| + \sum_{i=1}^{|\mathcal{A}|} P(t_i = c_j|a_i)} \tag{6}$$

The final expression for the posterior probabilities $P(c_j|a_i, \theta)$, i.e. the probability of the class given an article, is obtained via Bayes' Rule:

$$\begin{aligned} P(c_j|a_i, \hat{\theta}) &= \frac{P(c_j|\hat{\theta})P(a_i|c_j, \hat{\theta})}{P(a_i|\hat{\theta})} \\ &\propto P(c_j|\hat{\theta})P(a_i|c_j, \hat{\theta}) \end{aligned} \tag{7}$$

Expression 7 can be progressively expanded by replacing each one of the terms by the corresponding expressions, given in equations 2, 3, 5 and 6.

Despite its unrealistic simplifications and assumptions (data generated by mixture model; 1:1 correspondence between mixture components and topics; conditional independence of word events; article length independence), Naive Bayes classifiers have proven to provide fair classification performance [13], [14]. Due to their simplicity, these are well suited for text classification tasks, where the number of features is usually large (i.e. dictionary sizes often reaching orders of thousands of words [13]).

D. Semi-Supervised Learning via Expectation Maximization (EM)

The MNB text classifier described in Section III-C falls into the scope of supervised learning, only taking labeled data into account. Despite its fair performance when trained with large amounts of labeled data, Nigam et al. [14] notice how it suffers when faced with small-sized datasets and show some advantages (regarding classification accuracy) of expanding such models to the semi-supervised learning scope, i.e. considering both labeled and unlabeled data.

As with the case of the MNB classifier, the parameters θ are obtained by MAP estimation, i.e. maximizing expression 4. However, we should note that now the training dataset is composed by the labeled and unlabeled subsets $\mathcal{A}^\ell$ and $\mathcal{A}^u$. Following the Semi-Supervised GM ideas introduced in Section III-B, the expression for $P(\mathcal{A}|\theta)$ becomes:

$$\begin{aligned} P(\mathcal{A}|\theta) = & \prod_{a_i \in \mathcal{A}^u} \sum_{j=1}^{|\mathcal{C}|} P(c_j|\theta) P(a_i|c_j, \theta) \\ & \times \prod_{a_i \in \mathcal{A}^\ell} P(t_i = c_j|\theta) P(a_i|t_i = c_j, \theta) \end{aligned} \tag{8}$$

Note that for the set of labeled articles, $\mathcal{A}^\ell$, one already knows the true topic/mixture model component c_j which generated each article a_i, hence there is not the need of referring to all components in $\mathcal{C}$. However, for the unlabeled set $\mathcal{A}^u$, each component c_j has a contribution to the generation of each article a_i which must be taken into account.

As described in [14] (and as with the case of the MNB classifier), expression 8 can be passed to logarithmic form, with the maximization of $P(\theta)P(\mathcal{A}|\theta)$ being accomplished by solving the system of partial derivatives of $\log(P(\theta)P(\mathcal{A}|\theta))$. Here we use the same nomenclature as that used in [14], $\ell(\theta|\mathcal{A}) \equiv \log(P(\theta)P(\mathcal{A}|\theta))$:

$$\begin{aligned} \ell(\theta|\mathcal{A}) = & \log(P(\theta)) \\ & + \sum_{a_i \in \mathcal{A}^u} \log\left(\sum_{j=1}^{|\mathcal{C}|} P(c_j|\theta) P(a_i|c_j, \theta) \right) \\ & + \sum_{a_i \in \mathcal{A}^\ell} \log\left(P(t_i = c_j|\theta) P(a_i|t_i = c_j, \theta)\right) \end{aligned} \tag{9}$$

The log of sums over all mixture components $\mathcal{C}$ for all $a_i \in \mathcal{A}^u$ makes the problem computationally intractable [14]. This is where the EM algorithm comes into play, providing an iterative process to obtain a MAP estimation of the parameters θ, including the unlabeled data [14]. We now proceed with a brief description of the overall MNB + EM process, without detailing the inner works of EM (refer to [14] for further details):

1) Train a MNB classifier with the labeled dataset $\mathcal{A}^\ell$ only. Find the estimated parameters $\hat{\theta}$ using expressions 5 and 6.
2) **EM's E Step:** Use the classifier governed by the current $\hat{\theta}$ parameters to estimate the contribution of each mixture model component c_j to the generation of each article in the unlabeled dataset $\mathcal{A}^u$, i.e. determine $P(c_j|a_i, \hat{\theta})$ $\forall$ $a_i \in \mathcal{A}^u$ using expression 7.
3) **EM's M Step:** Re-estimate the $\hat{\theta}$ parameters at the light of the new $P(c_j|a_i, \hat{\theta})$ for both $\mathcal{A}^\ell$ and $\mathcal{A}^u$, using expressions 5 and 6. Note that now $P(c_j|a_i, \hat{\theta})$ varies between 0 and 1 for $\mathcal{A}^u$ (as opposed to $P(c_j|a_i, \hat{\theta}) \in \{0, 1\}$ $\forall$ $a_i \in \mathcal{A}^\ell$).
4) Evaluate $\ell(\theta|\mathcal{A})$ using expression 9. If $\Delta\ell(\theta|\mathcal{A}) > T$, T being a convergence threshold, return to step 2. Else, accept the classifier governed by the current $\hat{\theta}$ as the final solution.

Nigam et al. [14] note that while this simple MNB + EM combination performs well over datasets containing small amounts of labeled data vs. large amounts of unlabeled data (e.g. with differences within the range of 1000 vs. 10000 [14], in the case of the 20 Newsgroups dataset [18]), it may decrease the classification accuracy of an MNB classifier in the presence of large labeled datasets. As with other semi-supervised learning problems [19], these reductions in performance are due to violations of the model assumptions, previously stated in Sections III-B and III-C.

1) Extensions to the EM Algorithm: EM-λ: Nigam et al. [14] propose two extensions to the EM approach described in Section III-D which attempt to cope with violations of some MNB model assumptions: the EM-Multiple and EM-λ techniques. In the first case, the idea is to tackle violations of the 1:1 mixture component-to-topic correspondence assumption, allowing N:1 correspondences. The idea is that some topics may be separated into sub-topics (e.g. ‘football’ and ‘cricket’ in ‘sports’), with co-occurrences of words that may be better captured by multiple multinomial distributions. The unsupervised learning component of the problem is now detached from a particular number of ‘soft clusters’, which may be now determined via cross-validation [14].

We describe the second case – EM-λ – with more detail. Notice that when $|\mathcal{A}^u| \gg |\mathcal{A}^\ell|$, the influence of $\mathcal{A}^\ell$ in the maximization of expression 9 is negligible, i.e. EM will be essentially be performing unsupervised clustering [14]. The role of $\mathcal{A}^\ell$ would then be limited to provide the initial parameter estimates $\hat{\theta}$ and provide the number and topic correspondences for the ‘latent’ variables of the mixture model. This may result in poor classification accuracy if the distribution of the data does not follow the GM's assumptions.

One solution is to reduce the influence of the unlabeled data in expression 9 by a factor λ with $0 \le \lambda \le 1$. The difference between EM-λ and the method shown in Section III-D is in the M-Step, with equations 5 and 6 altered to include the λ factors:

$$\begin{aligned} \hat{\theta}_{w_k|c_j} & \equiv P(w_k|c_j, \hat{\theta}) \\ & = \frac{\alpha + \sum_{i=1}^{|\mathcal{A}|} \Lambda(i) N_{i,k} P(t_i = c_j|a_i)}{\alpha|\mathcal{D}| + \sum_{s=1}^{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{A}|} N_{i,s} P(t_i = c_j|a_i)} \end{aligned} \tag{10}$$

$$\hat{\theta}_{c_j} \equiv P(c_j|\hat{\theta}) = \frac{\alpha + \sum_{i=1}^{|\mathcal{A}|} \Lambda(i) P(t_i = c_j|a_i)}{\alpha|\mathcal{C}| + |\mathcal{A}^\ell| + \lambda|\mathcal{A}^u|} \tag{11}$$

where the function $\Lambda(i)$ defines the λ weighing factor to apply, whether the an article a_i belongs to $\mathcal{A}^\ell$ or $\mathcal{A}^u$:

$$\Lambda(i) = \begin{cases} \lambda & \text{if } a_i \in \mathcal{A}^u \\ 1 & \text{if } a_i \in \mathcal{A}^\ell \end{cases} \tag{12}$$

If we consider the extreme values allowed for λ, when $\lambda = 0$ the influence of the unlabeled data is mitigated, resulting in a MNB classifier; with $\lambda = 1$ the contribution of the unlabeled dataset is maximum, corresponding to the basic MNB + EM approach described in Section III-D. In [14], the value of λ is chosen as that which maximizes the leave-one-out cross-validation classification accuracy.

IV. Implementation

This section provides noteworthy details on our implementation of the methods introduced in Section III.

A. Multinomial Naive Bayes

The computation of the class conditional probabilities $P(a_i|c_j, \theta)$ via expression 3 involves the multiplication of $|\mathcal{D}|$ probabilities $P(w_k|c_j, \theta)$. Considering that text classification tasks work with dictionaries composed by thousands of words (e.g. $\sim$60000 for the 20 Newsgroups dataset [18]), the multiplication result may end up at zero due to floating point precision underflow. To overcome this issue, the same computations can be performed in the logarithmic domain, with expression 3 becoming:

$$\log(P(a_i|c_j, \theta)) \propto \sum_{k=1}^{|\mathcal{D}|} N_{i,k} \times \log(P(w_k|c_j, \theta)) \qquad (13)$$

B. Expectation Maximization (EM)

As we get the values of $P(c_j|a_i, \theta)$ in logarithmic form, the floating point underflow problem may also arise when computing the 'log-of-sums' component in expression 9. In order to circumvent it, we apply the 'Log-Sum-Exp' (LSE) trick, i.e. considering:

$$\begin{aligned} \log \sum_{j=1}^{|\mathcal{C}|} P(c_j|a_i, \theta) &= \log \sum_{j=1}^{|\mathcal{C}|} e^{\log(P(c_j|a_i,\theta))} \\ &= m + \log \sum_{j=1}^{|\mathcal{C}|} e^{\log(P(c_j|a_i,\theta)) - m} \end{aligned} \qquad (14)$$

where m is the maximum value of $\log(P(c_j|a_i, \theta))$, for each a_i. In our case we also set $P(c_j|a_i, \theta) = 0$ when $\log(P(c_j|a_i, \theta)) - m < p$, i.e. we discard the posteriors which, even after LSE, are still smaller than a threshold e^p, e.g. and therefore considered too small to impact the final result. We also use LSE after EM's E step, before applying expression 5 over $\mathcal{A}^u$.

We only run each EM's M step (see Section III), over the unlabeled data $\mathcal{A}^u$, since the $\hat{\theta}$ values for $\mathcal{A}^\ell$ are previously calculated in step 1 and do not change over the iterations (the respective $P(c_j|a_i, \hat{\theta})$ values do not change during the E step, as these are given).

V. Experiments

This section describes the experiments we conducted to evaluate our and third party implementation of the Multinomial Naive Bayes models and respective semi-supervised extensions described in Section III.

Partition	# Articles	# Unique Words
Training	11260	53485
Test	7502	60698

TABLE I: Characteristics of version of `20news-bydate` dataset used in experiments.

A. Experimental Setup

We start with a brief description of the dataset used in our experiments, followed by the plan of experiments whose results are shown in Section V-B.

1) Dataset: We work with the 20 Newsgroups dataset [18], which consists in a collection of approx. 19000 web forum posts, (almost) evenly separated across 20 different topics [3]. Although separated from each other, some of the 20 classes are related to each other as sub-topics of a larger category (e.g. `rec.sport.baseball` and `rec.sport.baseball` as sub-topics of `rec.sport.*`). The posts were collected over a period of several months in 1993 [14].

For our experiments, we consider a modified version of the `20news-bydate` dataset [4]. The `20news-bydate` provides a dataset sorted by date, with separate training and test subsets (test subset composed by later posts), comprising a total of 18846 newsgroups articles. Using the Rainbow tools [5], developed by the authors of [13], [14], we have modified the dataset to (1) remove newsgroups headers (a common practice since the article headers include the name of the newsgroup they belong to, which would make classification trivial), (2) remove 524 common 'stop words' (e.g. 'the', 'of') and (3) remove words which occur only 1 time. These feature selection approaches are made in previous works on text classification [13], [14], [16], [17] and are mainly used as attempts to reduce the data dimensionality (i.e. size of the dictionary, $|\mathcal{D}|$. We have noticed that (2) was necessary to achieve similar MNB accuracies to those reported in [13], [14]. The characteristics of the modified `20news-bydate` are summarized in Table I.

2) Experimental Protocol: The protocol is structured to test the different methods presented in Section III, in the same order.

A) Multinomial Naive Bayes

We test our implementation of the MNB classifier, together with Rainbow's, evaluating accuracy and confusion between actual and predicted classifications under different conditions. We evaluate total and per-class accuracy Acc in %, according to expression 15:

$$Acc = \frac{\text{\# correctly classified articles}}{\text{total \# classified articles}} \qquad (15)$$

The MNB classifiers are tested over the test set (see Table I), for different sample sizes

[3] A list of topics is available in http://qwone.com/~jason/20Newsgroups/.

[4] Available in http://qwone.com/~jason/20Newsgroups/.

[5] Available in http://www.cs.cmu.edu/~mccallum/bow/rainbow/. Due to compiling issues with the original versions of Rainbow, we have used the following patched version which compiles with `gcc` version 4.6.3: https://github.com/brendano/bow/.

$|\mathcal{A}^\ell| = \{20, 50, 100, 500, 1000, 5000, 7520\}$ from the training set. For each case, we randomly sample $|\mathcal{A}^\ell|/20$ articles from each of the 20 topics and show the average results over 20 runs of this procedure.

B) Multinomial Naive Bayes + EM

We test our implementation of the MNB + EM combination, together with Rainbow's, again evaluating accuracy and confusion between actual and predicted classifications under different conditions. The selection of $\mathcal{A}^\ell$ and $\mathcal{A}^u$ sets is done as follows:

1) We first select a $\mathcal{A}^\ell$ sample of size $|\mathcal{A}^\ell| = \{20, 40, 100, 300, 500, 800, 1000, 1260\}$ from the training subset, again $|\mathcal{A}^\ell|/20$ articles from each of the 20 topics.
2) We use the remaining $11260 - |\mathcal{A}^\ell|$ as unlabeled data $\mathcal{A}^u$.

The values of $|\mathcal{A}^\ell|$ are chosen so that $|\mathcal{A}^u \geq 10000|$, for comparison with the results from [14]. Again, we show the average results over 20 runs of this procedure. As a baseline for direct comparison we use the results for the fully supervised MNB classifier, trained with each different $\mathcal{A}^\ell$ set.

C) Multinomial Naive Bayes + EM-λ

We test our implementation of the MNB + EM-λ combination, in a similar way to that of the basic EM case. Here we additionally test several values of $\lambda = \{0.01, 0.02, 0.1, 0.2, 0.5, 1.0\}$, evaluating accuracy and confusion between actual and predicted classifications.

B. Experimental Results

We present the results for the experiments described in the previous section. Due to problems with our implementation of EM and EM-λ[6], we only compare our implementation of MNB in MATLAB to that of the Rainbow framework. Nevertheless, we provide the results obtained with Rainbow, using the modified version of the `20news-bydate` dataset.

In Figure 2 we present the results of the supervised MNB classifier. The accuracy values for the Rainbow's version closely follow those reported in [14]. For low values of $|A^\ell|$, our implementation of MNB (which follows the methodology provided in Section III-C) provides worse results than Rainbow's, with closer values for $|A^\ell| > 5000$.

The results for test B, shown in Figure 3, partly reproduce the results found in [14], since for the lower $|A^\ell|$ the average results of MNB + EM surpass those for MNB. For $|A^\ell| > 40$, MNB provides better accuracies, nevertheless one should note we use a larger testing set (7502 vs. 4000) and a different method for choosing A^ℓ and A^u. MNB + EM accuracy strongly varies during the 20 runs, consistently providing higher maximums and lower minimums than MNB, for all $|A^\ell|$ (see Figure 4). The maximum values for MNB + EM approach those reported in [14]. Figures 5 and 6 show four cases of confusion matrices for test B, $A^\ell = 1260$, for

[6]Errors in the M-step of the EM algorithm. The code is available in http://paginas.fe.up.pt/~up200400437/mlproject.zip for reference

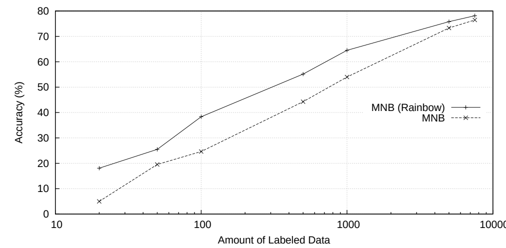

Fig. 2: Results for test A: MNB, using Rainbow's framework and our implementation in MATLAB.

both MNB and MNB + EM methods. One can clearly verify EM's clustering nature on the right side of Figure 4, with a significant number of false predictions for articles belonging to `comp.*` subgroups, mostly classified as belonging to the class `comp.graphics`. As expected, the main focus of confusion occur for clusters of subgroups, namely `comp.*`, `sci.*` (with a considerable number of articles belonging to several `comp.*` subgroups being misclassified as `sci.crypt`) and `talk.*`, for both MNB and MNB + EM cases.

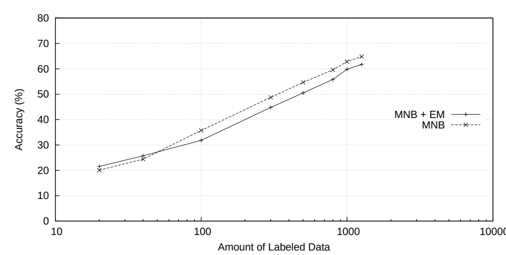

Fig. 3: Results for test B: MNB + EM, using Rainbow's framework. The results of MNB for the same A^ℓ are also given.

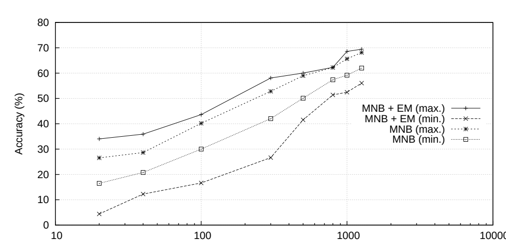

Fig. 4: Results for test B: MNB + EM, using Rainbow's framework. The results of MNB for the same A^ℓ are also given.

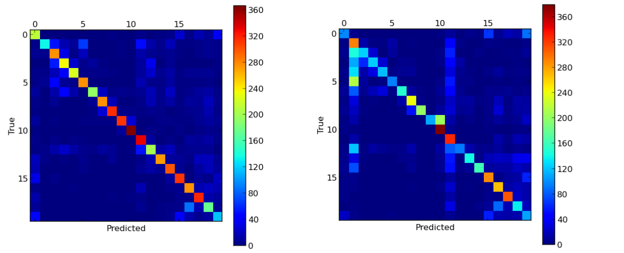

Fig. 5: Results for test B: Confusion matrices for best (68.12 %) and worst (61.99 %) results of MNB, $|A^\ell| = 1260$.

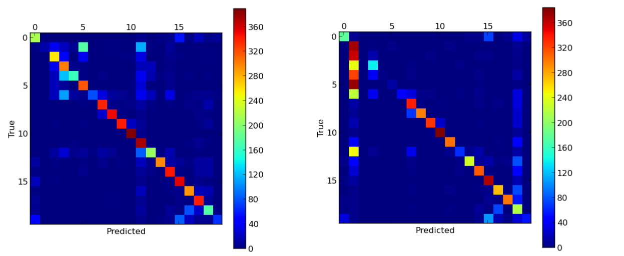

Fig. 6: Results for test B: Confusion matrices for best (left side, 69.44 %) and worst (right side, 56.04 %) results of MNB + EM, $|A^\ell| = 1260$.

Figure 7 shows the results for the EM-λ method, for several values of $L = 1/\lambda$[7]. We verified that for low values of $|A^\ell|$, lower values of L (and therefore higher λ) provide better results, with the top accuracy verified for $\lambda = 1$. As $|A^\ell|$ increases, higher accuracy values tend to be favored by lower values of λ. Nevertheless, the best results for EM-λ keep occurring for $\lambda = 1$, which seems somewhat counter-intuitive.

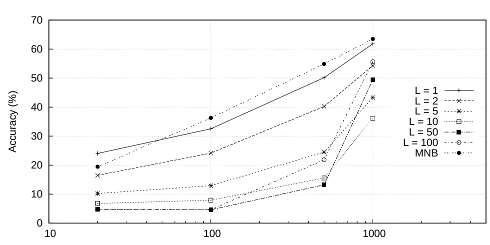

Fig. 7: Results for test C: MNB + EM-λ, using Rainbow's framework. The results of MNB for the same A^ℓ are also given.

VI. Conclusions

In this work we study the field of semi-supervised learning, applied to the field of text classification. We closely follow the work in [13], [14], reporting out understanding of the presented models.

[7]We use Rainbow's option `--em-unlabeled-normalizer`, which determines the number of unlabeled articles it takes to equal a labeled article.

We follow the methods presented by Nigam et al. [14] to implement our versions of the MNB and MNB + EM (including EM-λ) approaches. We take the well-known 20 Newsgroups dataset [18], particularly a version designated by `20news-bydate`, to evaluate the studied semi-supervised models in the context of a real-world dataset. The `20news-bydate` dataset is manipulated using the Rainbow tools — developed by the authors of the base literature reviewed in this paper — in order to pre-process it according to commonly used procedures and divide it into training and test partitions, also creating unlabeled samples for the semi-supervised setting.

Our version of MNB is compared with Rainbow's, producing lower accuracy values for small amounts of labeled data, approaching Rainbow's accuracy rates for higher amounts of labeled data ($|A^\ell| > 5000$). Due to problems with our implementation of MNB + EM, making its results unsuitable for comparison, we use Rainbow's implementation of MNB + EM and EM-λ over the `20news-bydate` dataset to experimentally validate the previously studied semi-supervised methods. We verify that, on average MNB + EM performs better than MNB for $|A^\ell| < 100$, with $11220 \le |A^u| \le 11240$. We also verify that, despite its large variations, for at least one of the test runs performed for every $|A^\ell|$, MNB + EM surpasses MNB. Regarding EM-λ, while smaller weights for unlabeled data seem to favor accuracy for larger $|A^\ell|$, the best results are always obtained for $\lambda = 1$. For $|A^\ell| \ge 1000$, the results seem intuitive, as the accuracy approaches that of MNB for smaller values of λ (i.e. smaller weights given to the unlabeled data).

References

[1] Andrea Passarella. A Survey on Content-Centric Technologies for the Current Internet: CDN and P2P solutions. *Computer Communications*, 35(1):1–32, 2012.

[2] M Handley. Why the Internet Only Just Works. *BT Technology Journal*, 24(3):119–129, 2006.

[3] J Pan, S Paul, and R Jain. A Survey of the Research on Future Internet Architectures. *Communications Magazine, IEEE*, 49(7):26–36, July 2011.

[4] George Xylomenos, Christopher N. Ververidis, Vasilios a. Siris, Nikos Fotiou, Christos Tsilopoulos, Xenofon Vasilakos, Konstantinos V. Katsaros, and George C. Polyzos. A Survey of Information-Centric Networking Research. *IEEE Communications Surveys & Tutorials*, PP(99):1–26, 2013.

[5] Teemu Koponen, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. A Data-Oriented (and Beyond) Network Architecture. In *ACM SIGCOMM Computer Communication Review*, volume 37, pages 181–193, 2007.

[6] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. Networking Named Content. *Proceedings of the 5th international conference on Emerging networking experiments and technologies CoNEXT 09*, 178(1):1–12, 2009.

[7] Dirk Trossen and George Parisis. Designing and Realizing an Information-Centric Internet. *IEEE Communications Magazine*, 50(7):60–67, 2012.

[8] Dipankar Raychaudhuri, Kiran Nagaraja, and Arun Venkataramani. MobilityFirst : A Robust and Trustworthy Mobility- Centric Architecture for the Future Internet. *ACM SIGMOBILE Mobile Computing and Communications Review*, 16(3):2–13, 2012.

[9] Dongsu Han, Peter Steenkiste, Michel Machado, and David G Andersen. XIA : Efficient Support for Evolvable Internetworking. In *The 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI'12)*, pages 1–14, 2012.

[10] Christian Dannewitz, Dirk Kutscher, BöRje Ohlman, Stephen Farrell, Bengt Ahlgren, and Holger Karl. Network of Information (NetInf) - An Information-Centric Networking Architecture. *Comput. Commun.*, 36(7):721–735, April 2013.

[11] J. K. Hedrick and A. Gira. *Control of Nonlinear Dynamic Systems: Theory and Applications*. 2010.

[12] Cheng Yi, Alexander Afanasyev, Lan Wang, Beichuan Zhang, and Lixia Zhang. Adaptive Forwarding in Named Data Networking. *ACM SIGCOMM Computer Communication Review*, 42(3):62–67, June 2012.

[13] Andrew McCallum and Kamal Nigam. A Comparison of Event Models for Naive Bayes Text Classification. In *AAAI-98 Workshop on Learning for Text Categorization*, pages 41–48. AAAI Press, 1998.

[14] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning - Special Issue on Information Retrieval*, 39(2-3):103–134, 2000.

[15] Jason D M Rennie, Lawrence Shih, Jaime Teevan, and David R Karger. Tackling the Poor Assumptions of Naive Bayes Text Classifiers. In *In Proceedings of the Twentieth International Conference on Machine Learning*, pages 616–623, 2003.

[16] Jiang Su, Jelber Sayyad-Shirabad, and Stan Matwin. Large Scale Text Classification using Semi-Supervised Multinomial Naive Bayes. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, pages 97–104, 2011.

[17] Ashraf M Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. Multinomial Naive Bayes for Text Categorization Revisited. In *Proceedings of the 17th Australian Joint Conference on Advances in Artificial Intelligence*, AI'04, pages 488–499, Berlin, Heidelberg, 2004. Springer-Verlag.

[18] Ken Lang. Newsweeder: Learning to Filter Netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339, 1995.

[19] X Zhu and A Goldberg. *Introduction to Semi-Supervised Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2009.