Summarization of Multi-Document Topic Hierarchies using Submodular Mixtures

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, 40076, India
bairi@cse.iitb.ac.in

Rishabh Iyer
University of Washington
Seattle, WA-98175, USA
rkiyer@u.washington.edu

Ganesh Ramakrishnan
IIT Bombay
Mumbai, 40076, India
ganesh@cse.iitb.ac.in

Jeff Bilmes
University of Washington
Seattle, WA-98175, USA
bilmes@uw.edu

Abstract

We study the problem of summarizing DAG-structured topic hierarchies over a given set of documents. Example applications include automatically generating Wikipedia disambiguation pages for a set of articles, and generating candidate multi-labels for preparing machine learning datasets (e.g., for text classification, functional genomics, and image classification). Unlike previous work, which focuses on clustering the set of documents using the topic hierarchy as features, we directly pose the problem as a submodular optimization problem on a topic hierarchy using the documents as features. Desirable properties of the chosen topics include document coverage, specificity, topic diversity, and topic homogeneity, each of which, we show, is naturally modeled by a submodular function. Other information, provided say by unsupervised approaches such as LDA and its variants, can also be utilized by defining a submodular function that expresses coherence between the chosen topics and this information. We use a large-margin framework to learn convex mixtures over the set of submodular components. We empirically evaluate our method on the problem of automatically generating Wikipedia disambiguation pages using human generated clusterings as ground truth. We find that our framework improves upon several baselines according to a variety of standard evaluation metrics including the Jaccard Index, F1 score and NMI, and moreover, can be scaled to extremely large scale problems.

1 Introduction

Several real world machine learning applications involve hierarchy based categorization of topics for a set of objects. Objects could be, e.g., a set of documents for text classification, a set of genes in functional genomics, or a set of images in computer vision. One can often define a natural topic hierarchy to categorize these objects. For example, in text and image classification problems, each document or image is assigned a hierarchy of labels — a baseball page is assigned the labels "baseball" and "sports." Moreover, many of these applications, naturally have an existing topic hierarchy generated on the entire set of objects (Rousu et al., 2006; Barutcuoglu et al., 2006; ling Zhang and hua Zhou, 2007; Silla and Freitas, 2011; Tsoumakas et al., 2010).

Given a DAG-structured topic hierarchy and a subset of objects, we investigate the problem of finding a subset of DAG-structured topics that are induced by that subset (of objects). This problem arises naturally in several real world applications. For example, consider the problem of identifying appropriate label sets for a collection of articles. Several existing text collection datasets such as 20 Newsgroup[1], Reuters-21578[2] work with a predefined set of topics. We observe that these topic names are highly abstract[3] for the articles categorized under them. On the other hand, techniques proposed by systems such as Wikipedia Miner (Milne, 2009) and TAGME (Ferragina and Scaiella, 2010) generate several labels for each article in the dataset that are highly specific to the article. Collating all labels from all articles to create a label

[1] http://qwone.com/~jason/20Newsgroups/

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[3] Topic *Concept* is more abstract than the topic *Science* which is more abstract than the topic*Chemistry*

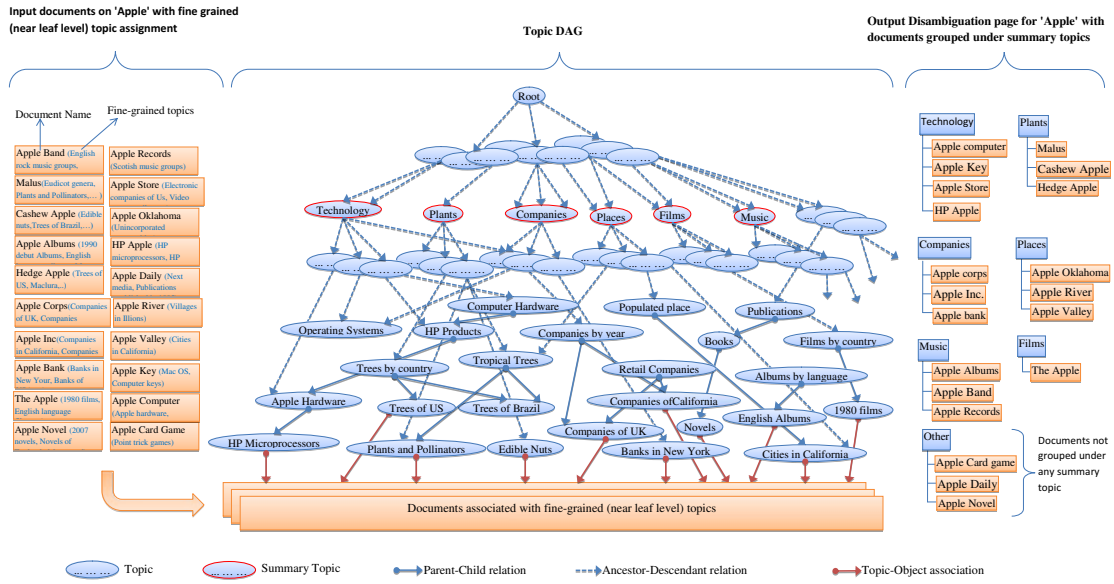

Figure 1: Topic Summarization overview. On the left, we show many documents related to Apple. In the middle, a Wikipedia category hierarchy shown as a topic DAG, links these documents at the leaf level. On the right, we show the output of our summarization process, which creates a set of summary topics (Plants, Technology, Companies, Films, Music and Places in this example) with the input documents classified under them.

set for the dataset can result in a large number of labels and become unmanageable. Our proposed techniques can summarize such large sets of labels into a smaller and more meaningful label sets using a DAG-structured topic hierarchy. This also holds for image classification problems and datasets like ImageNet (Deng et al., 2009). We use the term *summarize* to highlight the fact that the smaller label set semantically covers the larger label set. For example, the topics *Physics*, *Chemistry*, and *Mathematics* can be summarized into a topic *Science*.

A particularly important application of our work (and the one we use for our evaluations in Section 4) is the following: Given a collection of articles spanning different topics, but with similar titles, automatically generate a disambiguation page for those titles using the Wikipedia category hierarchy[4] as a topic DAG. Disambiguation pages[5] on Wikipedia are used to resolve conflicts in article titles that occur when a title is naturally associated with multiple articles on distinct topics. Each disambiguation page organizes articles into several groups, where the articles in each group pertain only to a specific topic. Disambiguations may be seen as paths in a hierarchy leading to different articles that arguably could have the same title. For example, the title *Apple*[6] can refer to a plant, a company, a film, a television show, a place, a technology, an album, a record label, and a newspaper daily. The problem then, is to organize the articles into multiple groups where each group contains articles of similar nature (topics) and has an appropriately discerned group heading. Figure 1 describes the topic summarization process for creation of the disambiguation page for "Apple".

All the above mentioned problems can be modeled as the problem of finding the most representative subset of topic nodes from a DAG-Structured topic hierarchy. We argue that many formulations of this problem are natural instances of submodular maximization, and provide a learning framework to create submodular mixtures to solve this problem. A set function $\mathrm{f}(.)$ is said to be submodular if for any element v and sets $A \subseteq B \subseteq V \setminus \{v\}$, where V represents the ground set of elements, $\mathrm{f}(A \cup \{v\}) - \mathrm{f}(A) \geq \mathrm{f}(B \cup \{v\}) - \mathrm{f}(B)$. This is called the diminishing returns property and states, informally, that adding an element to a smaller set increases the function value more than adding that element to a larger set. Submodular functions naturally model notions of coverage and diversity in applications, and therefore, a number of machine learning problems can be modeled as forms of submodular optimization (Kempe et al., 2003; Krause and Guestrin, 2005; Narasimhan and Bilmes, 2004; Iyer et al., 2013; Lin and Bilmes, 2012; Lin and Bilmes, 2010). In this paper, we investigate structured prediction methods for learn-

[4] http://en.wikipedia.org/wiki/Help:Categories

[5] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation

[6] http://en.wikipedia.org/wiki/Apple_(disambiguation)

ing weighted mixtures of submodular functions to summarize topics for a collection of objects using DAG-structured topic hierarchies. Throughout this paper we use the terms "topic" and "category" interchangeably.

1.1 Related Work

To the best of our knowledge, the specific problem we consider here is new. Previous work on identifying topics can be broadly categorized into one of the following types: a) cluster the objects and then identify names for the clusters; or b) dynamically identify topics (including hierarchical) for a set of objects. LDA (Blei et al., 2003) clusters the documents and simultaneously produces a set of topics into which the documents are clustered. In LDA, each document may be viewed as a mixture of various topics and the topic distribution is assumed to have a Dirichlet prior. LDA associates a group of high probability words to each identified topic. A name can be assigned to a topic by manually inspecting the words or using additional algorithms like (Mei et al., 2007; Maiya et al., 2013). LDA does not make use of existing topic hierarchies and correlation between topics. The Correlated Topic Model (Blei and Lafferty, 2006) induces a correlation structure between topics by using the logistic normal distribution instead of the Dirichlet. Another extension is the hierarchical LDA (Blei et al., 2004), where topics are joined together in a hierarchy by using the nested Chinese restaurant process. Nonparametric extensions of LDA include the Hierarchical Dirichlet Process (Teh et al., 2006) mixture model, which allows the number of topics to be unbounded and learnt from data and the Nested Chinese Restaurant Process which allows topics to be arranged in a hierarchy whose structure is learnt from data. In each of these approaches, unlike our proposed approach, an existing topic hierarchy is not used, nor is any additional object-topic information leveraged.

The pachinko allocation model (PAM)(Li and McCallum, 2006) captures arbitrary, nested, and possibly sparse correlations between topics using a DAG. The leaves of the DAG represent individual words in the vocabulary, while each interior node represents a correlation among its children, which may be words or other interior nodes (topics). PAM learns the probability distributions of words in a topic, subtopics in a topic, and topics in a document. We cannot, however, generate a subset of topics from a large existing topic DAG that can act as summary topics, using PAM.

HSLDA (Perotte et al., 2011) introduces a hierarchically supervised LDA model to infer hierarchical labels for a document. It assumes an existing label hierarchy in the form of a tree. The model infers one or more labels such that, if a label l is inferred as relevant to a document, then all the labels from l to the root of the tree are also inferred as relevant to the document. Our approach differs from HSLDA since: (1) we use the label hierarchy to infer a set of labels for a *group* of documents; (2) we do not enforce the label hierarchy to be a tree as it can be a DAG; and (3) generalizing HSLDA to use a DAG structured hierarchy and infer labels for a *group* of documents (e.g., combining into one big document) also may not help in solving our problem. HSLDA will apply *all* the relevant labels to the documents as per the classifier that it learns for every label. Moreover, the "root" label is always applied and it is very likely that many labels near the top level of the label hierarchy are also classified as relevant to the group of documents.

Wei and James (Bi and Kwok, 2011) present a hierarchical multi-label classification algorithm that can be used on both tree and DAG structured hierarchies. They formulate a search for the optimal consistent multi-label as the finding of the best subgraph in a tree/DAG. In our approach, we assume, individual documents are already associated with one or more topics and we find a consistent label set for a group of documents using the DAG structured topic hierarchy.

Medelyan et al. (Medelyan et al., 2008) and Ferragina et al. (Ferragina and Scaiella, 2010) detect topics for a document using Wikipedia article names and category names as the topic vocabulary. These systems are able to extract signals from a text document and identify Wikipedia articles and/or categories that optimally match the document and assign those article/category names as topics for the document. When run on a large collection of documents, these approaches generate enormous numbers of topics, a problem our proposed approach addresses.

1.2 Our Contributions

While most prior work discussed above focuses on the underlying set of documents, (*e.g.*, by clustering documents), we focus directly on the topics. In particular, we formulate the problem as subset selection on the set of topics within a DAG while simultaneously considering the documents to be categorized. Our method can scale to the colossal size of the DAG (1 million topics and 3 million correlation links between topics in Wikipedia). Moreover, our approach can

naturally incorporate outputs from many of the aforementioned algorithms. Our approach is based on submodular maximization and mixture learning, which has been successfully used in applications such as document summarization (Lin, 2012) and image summarization (Tschiatschek et al., 2014), but has never been applied to topic identification tasks or, more generally, DAG summarization.

We introduce a family of submodular functions to identify an appropriate set of topics from a DAG structured hierarchy of topics for a group of documents. We characterize this topic appropriateness through a set of desirable properties such as coverage, diversity, specificity, clarity, and relevance. Each of the submodular function components we consider are monotone, thereby ensuring a near optimal performance obtainable via a simple greedy algorithm for optimization.[7]. We also show how our technique naturally embodies outputs of other algorithms such as LDA, clustering, and classifications. Finally, we utilize a large margin formulation for learning mixtures of these submodular functions, and show how we can optimally learn them from training data.

Our approach demonstrates how to utilize the features collectively in the document space and the topic space to infer a set of topics. From an empirical perspective, we introduce and evaluate our approach on a dataset of around 8000 disambiguations that was extracted from Wikipedia and subsequently cleaned using the methods described in the experimentation section. We show that our learning framework outperforms many of the baselines, and is practical enough to be used on large corpora.

2 Problem Formulation

Let $G\,(V, E)$ be the DAG structured topic hierarchy with V topics. These topics are observed to have a parent child (*isa*) relationship forming a DAG. Let D be the set of documents that are associated with one or more of these topics. The middle portion of Figure 1 depicts a topic hierarchy with associated documents. The association links between the documents and topics can be hard or soft. In case of a hard link, a document is attached to a set of topics. Examples include multi-labeled documents. In case of a soft link, a document is associated with a topic with some degree of confidence (or probability). Furthermore, if a document is attached to a topic t, we assume that all the ancestor topics of t are also relevant for that document. This assumption has been employed in earlier works (Blei et al., 2004; Bi and Kwok, 2011; Rousu et al., 2006) as well. Given a budget of K, our objective is to choose a set of K topics from V, which best describe the documents in D. The notion of best describing topics is characterized through a set of desirable properties - coverage, diversity, specificity, clarity, relevance and fidelity - that K topics have to satisfy. The submodular functions that we introduce in the next section ensure these properties are satisfied. Formally, we solve the following discrete optimization problem:

$$S^* \in \underset{S \subseteq V : |S| \leq K}{\operatorname{argmax}} \sum_i w_i f_i(S) \qquad (1)$$

where, f_i are monotone submodular mixture components and $w_i \geq 0$ are the weights associated with those mixture components. Set S^* is the summary topics scored best.

It is easy to find massive (i.e., size in the order of million) DAG structured topic hierarchies in practice. Wikipedia's category hierarchy consists of more than 1M categories (topics) arranged hierarchically. In fact, they form a cyclic graph (Zesch and Gurevych, 2007). However, we can convert it to a DAG by eliminating the cycles as described in the supplementary material. YAGO (Suchanek et al., 2007) and Freebase (Bollacker et al., 2008) are other instances of massive topic hierarchies. The association of the documents with the existing topic hierarchy is also well studied. Systems such as WikipediaMiner (Milne, 2009), TAGME (Ferragina and Scaiella, 2010) and several annotation systems such as (Dill et al., 2003; Mihalcea and Csomai, 2007; Bunescu and Pasca, 2006) attach topics from Wikipedia (and other catalogs) to the documents by establishing the hard or soft links mentioned above.

Our goal is the following: Given a (ground set) collection V of topics organized in a pre-existing hierarchical DAG structure, and a collection D of documents, chose a size $K \in \mathbb{Z}_+$ representative subset of topics. Our approach is distinct from earlier work (e.g., (Kanungo et al., 2002; Blei et al., 2003)) where typically only a set of documents is classified and categorized in some way. We next provide a few definitions needed later in the paper.

Definition 1: Transitive Cover Γ): A topic t is said to cover a set of documents $\Gamma(t)$, called the transitive cover of the topic t, if for all documents $i \in \Gamma(t)$, either i is associated directly with topic t or with any of the descendant topics of t in the topic DAG. A natural extension of this definition to a set of topics T is defined as $\Gamma(T) = \cup_{t \in T} \Gamma(t)$.

[7] A simple greedy algorithm (Nemhauser et al., 1978) obtains a $1 - 1/e$ approximation guarantee for monotone submodular function maximization

Definition 2: Truncated Transitive Cover (Γ^α): This is a transitive cover of topic t, but with the limitation that the path length between a document and the topic t is not more than α. Hence, $|\Gamma^\alpha(t)| \leq |\Gamma(t)|$.

While our problem is closely related to clustering approaches, which consider the set of documents directly, there are some crucial differences. In particular, we focus on producing a clustering of documents where clusters are encouraged to honor a pre-defined DAG structured topic hierarchy. Existing agglomerative clustering algorithms focusing on the coverage of documents may not produce the desired clustering. To understand this, consider six documents d1, d2 ... d6 to be grouped into three clusters. There may be multiple ways to do this depending upon multiple aggregation paths present in the topic DAG: ((d1, d2), (d3, d4), (d5, d6)) or ((d1, d2, d3), (d4, d5), (d6)) or ((d1, d2, d3, d4), (d5), (d6)) or something else. Hence, we need more stringent measures to prefer one clustering over the others. Our work addresses this with a variety of quality criteria (coverage, diversity, specificity, clarity, relevance and fidelity, which are explained later in this paper) that are organically derived from well established submodular functions. And, most importantly, we learn the right mixture of these qualities to be enforced from the data itself. Furthermore, our approach also generalizes these clustering approaches, since one of the components in our mixture of submodular functions is defined via these unsupervised approaches, and maps a given clustering to a set of topics in the hierarchy.

3 Submodular Components and Learning

Summarization is the task of extracting information from a source that is both small in size but still representative. Our problem is different from traditional summarization tasks since we have an underlying DAG as a topic hierarchy that we wish to summarize in response to a subset of documents. Thus, a critical part of our problem is to take the graph structure into account while creating the summaries. Below, we identify properties we wish our summaries to posses.

Coverage: A summary set of topics should cover most of the documents. A document is said to be covered by a topic if there exists a path from the topic, going through intermediary descendant topics, to the document, i.e., the document is within the transitive cover of the topic.

Diversity: Summaries should be as diverse as possible, i.e., each summary topic should cover a unique set of documents. When a document is covered by more than one topic, that document is redundantly covered, e.g., "Finance" and "Banking" would be unlikely members of the same summary.

Summary qualities also involve "quality" notions, including:

Specificity/Clarity/Relevance/Coherence: These quality measures help us choose a set of topics that are neither too abstract nor overly specific. They ensure that the topics are clear and relevant to the documents that they represent. When additional information such as clustering (from LDA or other sources) and tagging (manual) documents is available, these quality criteria encourage the chosen topics to show resemblance (coherence) to those clustering/tagging in terms of transitive cover of documents they produce.

In the below, we define a variety of submodular functions that capture the above properties, and we then describe a large margin learning framework for learning convex mixtures of such components.

3.1 Submodular Components

3.1.1 Coverage Based Functions

Coverage components capture "coverage" of a set of documents.

Weighted Set Cover Function: Given a set of categories, $S \subseteq V$, define $\Gamma(S)$ as the set of documents covered — for each topic $s \in S$, $\Gamma(s) \subseteq D$ represents the documents covered by topic s and $\Gamma(S) = \cup_{s \in S} \Gamma(s)$. The weighted set cover function, defined as $f(S) = \sum_{d \in \Gamma(S)} w_d = w(\Gamma(S))$, assigns weights to the documents based on their relative importance (e.g., in Wikipedia disambiguation, the different documents could be ranked based on their priority).

Feature-based Functions: This class of function represents coverage in feature space. Given a set of categories $S \subseteq V$, and a set of features U, define $m_u(S)$ as the score associated with the set of categories S for feature $u \in U$. The feature set could represent, for example, the documents, in which case $m_u(S)$ represents the number of times document u is covered by the set S. U could also represent more complicated features. For example, in the context of Wikipedia disambiguation, U could represent TFIDF features over the documents. Feature based are then defined as $f(S) = \sum_{u \in U} \psi(m_u(S))$, where ψ is a concave (e.g., the square root) function. This function class has been successfully used in several applications (Kirchhoff and Bilmes, 2014; Wei et al., 2014a; Wei et al., 2014b).

3.1.2 Similarity based Functions

Similarity functions are defined through a similarity matrix $\mathbf{S} = \{s_{ij}\}_{i,j \in V}$. Given categories $i, j \in V$, similarity s_{ij} in our case can be defined as $s_{ij} = |\Gamma(i) \cap \Gamma(j)|$, i.e the number of documents commonly covered by both i and j.

Facility Location: The facility location function, defined as $f(S) = \sum_{i \in V} \max_{j \in S} s_{ij}$, is a natural model for k-medoids and exemplar based clustering, and has been used in several summarization problems (Tschiatschek et al., 2014; Wei et al., 2014a).

Penalty based diversity: A similarity matrix may be used to express a form of coverage of a set S but that is then penalized with a redundancy term, as in the following difference: $f(S) = \sum_{i \in V, j \in S} s_{ij} - \lambda \sum_{i \in S} \sum_{j \in S,} s_{i,j}$ (Lin and Bilmes, 2011)). Here $\lambda \in [0, 1]$. This function is submodular, but is not in general monotone, and has been used in document summarization (Lin and Bilmes, 2011), as a dispersion function (Borodin et al., 2012), and in image summarization (Tschiatschek et al., 2014).

3.1.3 Quality Control (QC) Functions

QC functions ensure a quality criteria is met by a set S of topics. We define the quality score of the set S as $F_q(S) = \sum_{s \in S} f_q(s)$, where $f_q(s)$ is the quality score of topic s for quality q. Therefore, $F_q(S)$ is a modular function in S. We investigate three types of quality control functions: Topic Specificity, Topic Clarity, and Topic Relevance.

Topic Specificity: The farther a topic is from the root of the DAG, the more specific it becomes. Topics higher up in the hierarchy are abstract and less specific. We therefore prefer topics low in the DAG, but lower topics also have less coverage. We define $f_{\text{specificity}}(s) = s_h$ where s_h is the height of topic s in the DAG. The root topic has height zero and the "height" increases as we move down the DAG in Figure 1.

Topic Clarity: Topic clarity is the fraction of descendant topics that cover one or more documents. If a topic has many descendant topics that do not cover any documents, it has less clarity. Formally, $f_{\text{clarity}}(s) = \frac{\sum_{t \in \text{descendants}(s)} [\![\Gamma(t) > 0]\!]}{|\text{descendants}(s)|}$, where $[\![.]\!]$ is the indicator function.

Topic Relevance: A topic is considered to be better related to a document if the number of hops needed to reach the document from that topic is lower. Given any set $A \subseteq D$ of document, and any topic $s \in V$, we can define $f_{\text{relevance}}(s|A) = \text{argmin}_\alpha \{\alpha : A \subseteq \Gamma^\alpha(s)\}$.

QC Functions As Barrier Modular Mixtures: We introduce a modular function for every QC function as follows

$$f^\alpha_{\text{specificity}}(s) = \begin{cases} 1 & \text{if the height of topic } s \text{ is at least } \alpha \\ 0 & \text{otherwise} \end{cases}$$

for every possible value of α. This creates a submodular mixture with as many components as the number of possible values of α. In our experiments with Wikipedia, we had α varying from 1 to 120 stepping by 1, adding 120 modular mixture components. Similarly, we define,

$$f^\beta_{\text{clarity}}(s) = \begin{cases} 1 & \text{if the clarity of topic } s \text{ is at least } \beta \\ 0 & \text{otherwise} \end{cases}$$

for every possible (discretized to make it countably finite) value of β. And,

$f^\gamma_{\text{relevance}}(s) = f_{\text{cov}}(s|\Gamma^\gamma(s))$, where $f_{\text{cov}}(.)$ is the coverage submodular function and $s|X$ indicates coverage of a topic s over a set of documents X. All these functions (modular and submodular terms) are added as mixture components in our learning framework to learn suitable weights for them. We then use these weights in our inference procedure to obtain a subset of topics as described in 3.2. We show from our experiments that this approach performs better than all other approaches and baselines.

3.1.4 Fidelity Functions

A function representing the fidelity of a set S to another reference set R is one that gets a large value when the set S represents the set R. Such a function scores inferred topics high when it resembles a reference set of topics and/or item clusters. The reference set in this case can be produced from other algorithms such as k-means, LDA and its variants or from a manually tagged corpus. Next we describe one such fidelity function.

Topic Coherence: This function scores a set of topics S high when the transitive cover (Definition 1) produced by the topics in S resembles the clusters of documents produced by an external source (k-means, LDA or manual). Given an external source that clusters the documents, producing T clusters $L_1, L_2, ..., L_T$ (for T topics), topic coherence is defined as: $f(S) = \sum_{t \in T} \max_{k \in S} w_{k,t}$ where $w_{k,t} = \text{harmonic\_mean}(w^p_{k,t}, w^r_{k,t})$ and $w^p_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|\Gamma(k)|}$ and $w^r_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|L_t|}$. Note that, $w^p_{k,t} \geq 0$ and $w^r_{k,t} \geq 0$ are the precision on recall of the resemblance and $w_{k,t}$ is the F1 measure. If the transitive cover of topics in S resembles the reference clusters L_t exactly, we attain maximum coherence (or fidelity). As the resemblance diminishes, the score decreases. The above function $f(S)$ is monotone submodular.

3.1.5 Mixture of Submodular Components:

Given the different classes of submodular functions above, we construct our submodular scoring functions $F_w(\cdot)$ as a convex combinations of these different submodular functions $f_1, f_2, \ldots, f_m$, above. In other words,

$$F_w(S) = \sum_{i=1}^{m} w_i f_i(S), \tag{2}$$

where $w = (w_1, \ldots, w_m)$, $w_i \geq 0, \sum_i w_i = 1$. The components f_i are submodular and assumed to be normalized: i.e., $f_i(\emptyset) = 0$, and $f_i(V) = 1$ for monotone functions and $\max_{A \subseteq V} f_i(A) \leq 1$ for non-monotone functions. A simple way to normalize a monotone submodular function is to define the component as $f_i(S)/f_i(V)$. This ensures that the components are *compatible* with each other. Obviously, the merit of the scoring function $F_w(\cdot)$ depends on the selection of the components.

3.2 Large Margin Learning

We optimize the weights w of the scoring function $F_w(\cdot)$ in a large-margin structured prediction framework. In this setting, we assume we have training data in the form of pairs of a set of documents, and a human generated summary as a set of topics. For example, in the case of Wikipedia disambiguation, we use the human generated disambiguation pages as the ground truth summary. We represent the set of ground-truth summaries as $\mathcal{S} = \{S_1, S_2, \cdots, S_N\}$. In large margin training, the weights are optimized such that ground-truth summaries $\mathcal{S}$ are separated from competitor summaries by a loss-dependent margin:

$$F_w(S) \geq F_w(S') + \mathcal{L}(S'), \quad \forall S \in \mathcal{S}, S' \in \mathcal{Y} \setminus \mathcal{S}, \tag{3}$$

where $\mathcal{L}(\cdot)$ is the loss function, and where $\mathcal{Y}$ is a structured output space (for example $\mathcal{Y}$ is the set of summaries that satisfy a certain budget B, i.e., $\mathcal{Y} = \{S' \subseteq V : |S'| \leq B\}$). We assume the loss to be normalized, $0 \leq \mathcal{L}(S') \leq 1, \forall S' \subseteq V$, to ensure that mixture and loss are calibrated. Equation (3) can be stated as $F_w(S) \geq \max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right], \forall S \in \mathcal{S}$ which is called *loss-augmented inference*. We introduce slack variables and minimize the regularized sum of slacks (Lin and Bilmes, 2012):

$$\min_{w \geq 0, \|w\|_1 = 1} \quad \sum_{S \in \mathcal{S}} \left[\max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right] - F_w(S)\right] + \frac{\lambda}{2}\|w\|_2^2, \tag{4}$$

where the non-negative orthant constraint, $w \geq 0$, ensures that the final mixture is submodular. Note a 2-norm regularizer is used on top of a 1-norm constraint $\|w\|_1 = 1$ which we interpret as a prior to encourage higher entropy, and thus more diverse mixture distributions. Tractability depends on the choice of the loss function. The parameters w are learnt using stochastic gradient descent as in (Tschiatschek et al., 2014).

3.3 Loss Functions

A natural choice of loss functions for our case can be derived from cluster evaluation metrics. Every inferred topic s induces a subset of documents, namely the transitive cover $\Gamma(s)$ of s. We compare these clusters with the clusters induced from the true topics in the training set and compute the loss.

In this paper, we use the Jaccard Index (JI) as a loss function. Let S be the inferred topics and T be the true topics. The Jaccard loss is defined as $\mathcal{L}_{\text{jaccard}}(S,T) = 1 - \frac{1}{k}\sum_{s \in S} \max_{t \in T} \frac{|\Gamma(s) \cap \Gamma(t)|}{|\Gamma(s) \cup \Gamma(t)|}$, where $k = |S| = |T|$ is the number of topics. When the clustering produced by the inferred and the true topics are similar, Jaccard loss is 0. When they are completely dissimilar, the loss is maximum, i.e., 1. Jaccard loss is a modular function.

3.4 Inference Algorithm: Greedy

Having learnt the weights for the mixture components, the resulting function $F_w(S) = \sum_{i=1}^{m} w_i f_i(S)$ is a submodular function. In the case when the individual components are themselves monotone (all our functions in fact are), $F_w(S)$ can be optimized by the accelerated greedy algorithm (Minoux, 1978). Thanks to submodularity, we can obtain near optimal solutions very efficiently. In case the functions are all monotone submodular, we can guarantee that the solution is within $1 - 1/e$ factor from the optimal solution.

4 Experimental Results

To validate our approach, we make use of Wikipedia category structure as a topic DAG and apply our technique to the task of automatic generation of Wikipedia disambiguation pages. We pre-processed the category graph to eliminate the cycles in order to make it a DAG. Each Wikipedia disambiguation page is manually created by Wikipedia editors by grouping a collection of Wikipedia articles into several groups. Each group is then assigned a name, which serves as a topic for the group. Typically, a disambiguation page segregates around 20-30 articles into 5-6 groups. Our goal is to measure how accurately we can recreate the groups for a disambiguation page and label them, given only the collection of articles mentioned in that disambiguation page (when actual groupings and labels are hidden).

4.1 Datasets

We parsed the contents of Wikipedia disambiguation pages and extracted disambiguation page names, article groups and group names. We collected about 8000 disambiguation pages that had at least four groups on them. Wikipedia category structure is used as the topic DAG. We eliminated few administrative categories such as "Hidden Categories", "Articles needing cleanup", and the like. The final DAG had about 1M topics and 3M links.

4.2 Evaluation Metrics

Every group of articles on the Wikipedia disambiguation page is assigned a name by the editors. Unfortunately, these names may not correspond to the Wikipedia category (topic) names. For example, one of the groups on the "Matrix" disambiguation page has a name "Business and government" and there is no Wikipedia category by that name. However, the group names generated by our (and baseline) method are from the Wikipedia categories (which forms our topic DAG). In addition, there can be multiple relevant names for a group. For example, a group on a disambiguation page may be called "Calculus", but an algorithm may rightly generate "Vector Calculus". Hence we cannot evaluate the accuracy of an algorithm just by matching the generated group names to those on the disambiguation page. To alleviate this problem, we adopt cluster-based evaluation metrics. We treat every group of articles generated by an algorithm under a topic for a disambiguation page as a cluster of articles. These are considered as *inferred* clusters for a disambiguation page. We compare them against the actual grouping of articles on the Wikipedia disambiguation page by treating those groups as *true* clusters. We can now adopt Jaccard Index, F1-measure, and NMI (Normalized Mutual Information) based cluster evaluation metrics described in (Manning et al., 2008). For each disambiguation page in the test set, we compute every metric score and then average it over all the disambiguation pages.

4.3 Methods Compared

We validated our approach by comparing against several baselines described below. We also compared two variations of our approach as described next. For each of these cases (baselines and two variations) we generated and compared the metrics (Jaccard Index, F1-measure and NMI) as described in the previous section.

KM_{docs}: K-Means algorithm run on articles as TF-IDF vectors of words. The number of clusters K is set to the number of true clusters on the Wikipedia disambiguation page.

$\text{KMed}_{\text{docs}}$: K-Medoids algorithm with articles as TF-IDF vectors of words. The number of clusters are set as in KM_{docs}.

$\text{KMed}_{\text{topics}}$: K-Medoids run on topics as TF-IDF vectors of words. The words for each topic is taken from the articles that are in the transitive cover of the topic.

LDA_{docs}: LDA algorithm with the number of topics set to the number of true clusters on the Wikipedia disambiguation page. Each article is then grouped under the highest probability topic.

SMML_{cov}: This is the submodular mixture learning case explained in section 3.1.5. Here we consider a mixture of all the submodular functions governing coverage, diversity, fidelity and QC functions. However, we exclude the similarity based functions described in section 3.1.2. Coverage based functions have a time complexity of $O(n)$ whereas similarity based functions are $O(n^2)$. By excluding similarity based functions, we can compare the quality of the results with and without $O(n^2)$ functions. We learn the mixture weights from the training set and use them during inference on the test set to subset K topics through the submodular maximization (Equation 1).

$\text{SMML}_{\text{cov+sim}}$: This case is similar to SMML_{cov} except that, we include similarity based submodular mixture components. This makes the inference time complexity $O(n^2)$.

We do not compare against HSLDA, PAM and few other techniques cited in the related work sections because they do not produce a subset of K summary topics — these are not directly comparable with our work.

4.4 Evaluation Results

We show that the submodular mixture learning and maximization approaches, i.e., SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other approaches in various metrics. In all these experiments, we performed 5 fold cross validation to learn the parameters from 80% of the disambiguation pages and evaluated on the rest of the 20%, in each fold.

In Figure 2a we summarize the results of the comparison of the methods mentioned above on Jaccard Index, F1 measure and NMI. Our proposed techniques SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other techniques consistently.

In Figures 2b and 2c we measure the number of test instances (i.e., disambiguation queries) in which each of the algorithms dominate (win) in evaluation metrics. In 60% of the disambiguation queries, SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ approaches

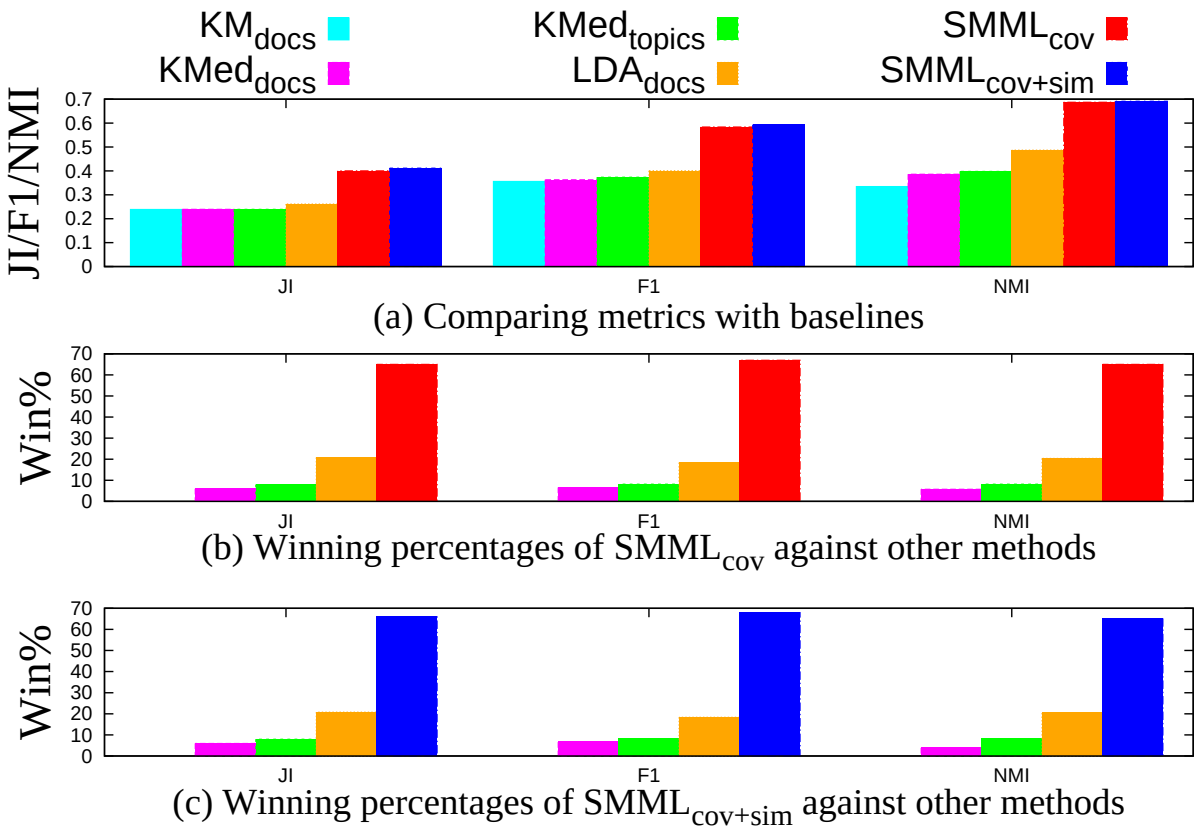

Figure 2: Comparison of techniques

produce higher JI, F1 and NMI than all other methods. This indicates that the clusters of articles produced by our technique resembles the clusters of articles present on the disambiguation page better than other techniques.

From Figures 2b and 2c it is clear that $O(n)$ time complexity based submodular mixture functions (SMML_{cov}) perform on par with $O(n^2)$ based functions ($\text{SMML}_{\text{cov+sim}}$), but at a greatly reduced execution time, demonstrating the sufficiency of $O(n)$ functions for our task. On the average, for each disambiguation query, SMML_{cov} took around 40 seconds (over 1M topics and 3M edges DAG) to infer the topics, whereas $\text{SMML}_{\text{cov+sim}}$ took around 35 minutes. Both these experiments were carried on a machine with 32 GB RAM, Eight-Core AMD Opteron(tm) Processor 2427.

5 Conclusions

We investigated a problem of summarizing topics over a massive topic DAG such that the summary set of topics produced represents the objects in the collection. This representation is characterized through various classes of submodular (and monotone) functions that captured coverage, similarity, diversity, specificity, clarity, relevance and fidelity of the topics. Currently we assume that the number of topics K is given as an input to our algorithm. It would be an interesting future problem to estimate the value of K automatically in our setting. As future work, we also plan to extend our techniques to produce a hierarchical summary of topics and scale it across heterogeneous collection of objects (from different domains) to bring all of them under the same topic DAG and investigate interesting cases thereon.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1162606, and by a Google, a Microsoft, and an Intel research award. Rishabh Iyer acknowledges support from the Microsoft Research Ph.D Fellowship.

References

Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. 2006. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, April.

W. Bi and J. T. Kwok. 2011. Multi-label classification on tree-and DAG-structured hierarchies. In *ICML*. ICML.

David M. Blei and John D. Lafferty. 2006. Correlated topic models. In *In Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. MIT Press.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, page 2003. MIT Press.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of Principles of Database Systems*, pages 155–166. ACM.

Razvan Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06), Trento, Italy*, pages 9–16, April.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. 2003. Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 178–186, New York, NY, USA. ACM.

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA.

R. Iyer, S. Jegelka, and J. Bilmes. 2013. Fast semidifferential-based submodular function optimization. ICML.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, Angela Y. Wu, Senior Member, and Senior Member. 2002. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:881–892.

D. Kempe, J. Kleinberg, and E. Tardos. 2003. Maximizing the spread of influence through a social network. In *SIGKDD*.

Katrin Kirchhoff and Jeff Bilmes. 2014. Submodularity for data selection in machine translation. October.

A. Krause and C. Guestrin. 2005. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of Uncertainity in Artificial Intelligence*. UAI.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 577–584, New York, NY, USA. ACM.

H. Lin and J. Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL*.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *The 49th Meeting of the Assoc. for Comp. Ling. Human Lang. Technologies (ACL/HLT-2011)*, Portland, OR, June.

H. Lin and J. Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, page 479490.

Hui Lin. 2012. *Submodularity in Natural Language Processing: Algorithms and Applications*. Ph.D. thesis, University of Washington, Dept. of EE.

Min ling Zhang and Zhi hua Zhou. 2007. Ml-knn: A lazy learning approach to multi-label learning. *PATTERN RECOGNITION*, 40:2007.

Arun S. Maiya, John P. Thompson, Francisco Loaiza-Lemos, and Robert M. Rolfe. 2013. Exploratory analysis of highly heterogeneous document collections. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1375–1383, New York, NY, USA. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Olena Medelyan, Ian H. Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the Wikipedia and AI workshop at AAAI-08*. AAAI.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 490–499, New York, NY, USA. ACM.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA. ACM.

David Milne. 2009. An open-source toolkit for mining wikipedia. In *In Proc. New Zealand Computer Science Research Student Conf*, page 2009.

Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, chapter 27, pages 234–243. Springer Berlin Heidelberg, Berlin/Heidelberg.

Mukund Narasimhan and Jeff Bilmes. 2004. PAC-learning bounded tree-width graphical models. In *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference (UAI-2004)*. Morgan Kaufmann Publishers, July.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294.

Adler J. Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. 2011. Hierarchically supervised latent dirichlet allocation. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *NIPS*, pages 2609–2617.

Juho Rousu, Craig Saunders, Sndor Szedmk, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626.

Jr. Silla, CarlosN. and AlexA. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Sebastian Tschiatschek, Rishabh Iyer, Hoachen Wei, and Jeff Bilmes. 2014. Learning Mixtures of Submodular Functions for Image Collection Summarization. In *Neural Information Processing Systems (NIPS)*.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer US.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2014a. Fast multi-stage submodular maximization. In *ICML*.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, Chris Bartels, and Jeff Bilmes. 2014b. Submodular subset selection for large-scale speech training data. *Proceedings of ICASSP, Florence, Italy*.

Torsten Zesch and Iryna Gurevych. 2007. Analysis of the wikipedia category graph for nlp applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, pages 1–8, Rochester, April. Association for Computational Linguistics.

Summarization of Multi-Document Topic Hierarchies using Submodular Mixtures

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, 40076, India
bairi@cse.iitb.ac.in

Rishabh Iyer
University of Washington
Seattle, WA-98175, USA
rkiyer@u.washington.edu

Ganesh Ramakrishnan
IIT Bombay
Mumbai, 40076, India
ganesh@cse.iitb.ac.in

Jeff Bilmes
University of Washington
Seattle, WA-98175, USA
bilmes@uw.edu

Abstract

We study the problem of summarizing DAG-structured topic hierarchies over a given set of documents. Example applications include automatically generating Wikipedia disambiguation pages for a set of articles, and generating candidate multi-labels for preparing machine learning datasets (e.g., for text classification, functional genomics, and image classification). Unlike previous work, which focuses on clustering the set of documents using the topic hierarchy as features, we directly pose the problem as a submodular optimization problem on a topic hierarchy using the documents as features. Desirable properties of the chosen topics include document coverage, specificity, topic diversity, and topic homogeneity, each of which, we show, is naturally modeled by a submodular function. Other information, provided say by unsupervised approaches such as LDA and its variants, can also be utilized by defining a submodular function that expresses coherence between the chosen topics and this information. We use a large-margin framework to learn convex mixtures over the set of submodular components. We empirically evaluate our method on the problem of automatically generating Wikipedia disambiguation pages using human generated clusterings as ground truth. We find that our framework improves upon several baselines according to a variety of standard evaluation metrics including the Jaccard Index, F1 score and NMI, and moreover, can be scaled to extremely large scale problems.

1 Introduction

Several real world machine learning applications involve hierarchy based categorization of topics for a set of objects. Objects could be, e.g., a set of documents for text classification, a set of genes in functional genomics, or a set of images in computer vision. One can often define a natural topic hierarchy to categorize these objects. For example, in text and image classification problems, each document or image is assigned a hierarchy of labels — a baseball page is assigned the labels "baseball" and "sports." Moreover, many of these applications, naturally have an existing topic hierarchy generated on the entire set of objects (Rousu et al., 2006; Barutcuoglu et al., 2006; ling Zhang and hua Zhou, 2007; Silla and Freitas, 2011; Tsoumakas et al., 2010).

Given a DAG-structured topic hierarchy and a subset of objects, we investigate the problem of finding a subset of DAG-structured topics that are induced by that subset (of objects). This problem arises naturally in several real world applications. For example, consider the problem of identifying appropriate label sets for a collection of articles. Several existing text collection datasets such as 20 Newsgroup[1], Reuters-21578[2] work with a predefined set of topics. We observe that these topic names are highly abstract[3] for the articles categorized under them. On the other hand, techniques proposed by systems such as Wikipedia Miner (Milne, 2009) and TAGME (Ferragina and Scaiella, 2010) generate several labels for each article in the dataset that are highly specific to the article. Collating all labels from all articles to create a label

[1] http://qwone.com/~jason/20Newsgroups/

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[3] Topic *Concept* is more abstract than the topic *Science* which is more abstract than the topic*Chemistry*

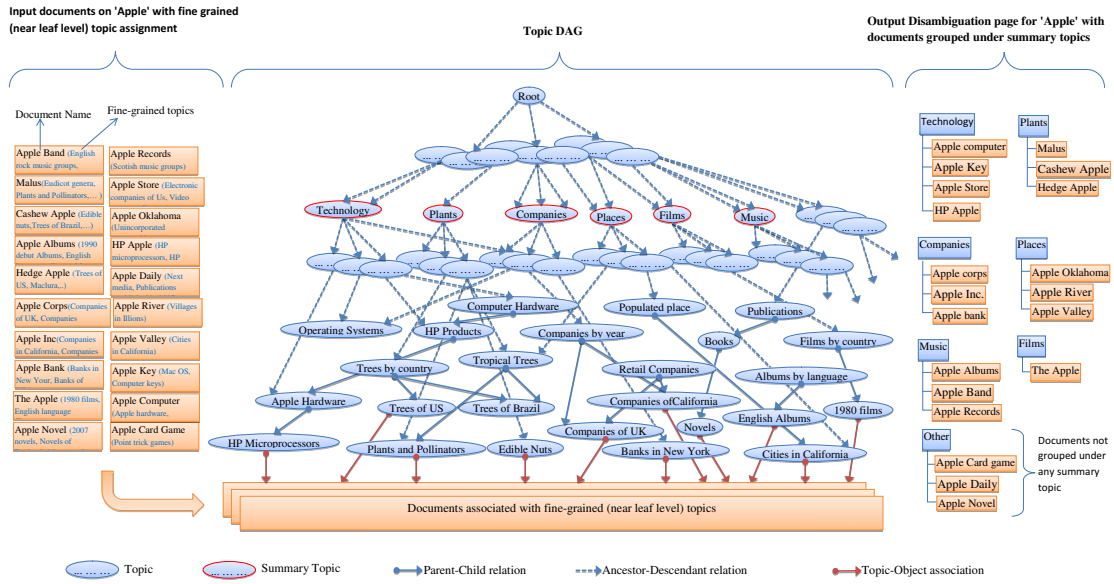

Figure 1: Topic Summarization overview. On the left, we show many documents related to Apple. In the middle, a Wikipedia category hierarchy shown as a topic DAG, links these documents at the leaf level. On the right, we show the output of our summarization process, which creates a set of summary topics (Plants, Technology, Companies, Films, Music and Places in this example) with the input documents classified under them.

set for the dataset can result in a large number of labels and become unmanageable. Our proposed techniques can summarize such large sets of labels into a smaller and more meaningful label sets using a DAG-structured topic hierarchy. This also holds for image classification problems and datasets like ImageNet (Deng et al., 2009). We use the term *summarize* to highlight the fact that the smaller label set semantically covers the larger label set. For example, the topics *Physics*, *Chemistry*, and *Mathematics* can be summarized into a topic *Science*.

A particularly important application of our work (and the one we use for our evaluations in Section 4) is the following: Given a collection of articles spanning different topics, but with similar titles, automatically generate a disambiguation page for those titles using the Wikipedia category hierarchy[4] as a topic DAG. Disambiguation pages[5] on Wikipedia are used to resolve conflicts in article titles that occur when a title is naturally associated with multiple articles on distinct topics. Each disambiguation page organizes articles into several groups, where the articles in each group pertain only to a specific topic. Disambiguations may be seen as paths in a hierarchy leading to different articles that arguably could have the same title. For example, the title *Apple*[6] can refer to a plant, a company, a film, a television show, a place, a technology, an album, a record label, and a newspaper daily. The problem then, is to organize the articles into multiple groups where each group contains articles of similar nature (topics) and has an appropriately discerned group heading. Figure 1 describes the topic summarization process for creation of the disambiguation page for "Apple".

All the above mentioned problems can be modeled as the problem of finding the most representative subset of topic nodes from a DAG-Structured topic hierarchy. We argue that many formulations of this problem are natural instances of submodular maximization, and provide a learning framework to create submodular mixtures to solve this problem. A set function $\mathrm{f}(.)$ is said to be submodular if for any element v and sets $A \subseteq B \subseteq V \setminus \{v\}$, where V represents the ground set of elements, $\mathrm{f}(A \cup \{v\}) - \mathrm{f}(A) \geq \mathrm{f}(B \cup \{v\}) - \mathrm{f}(B)$. This is called the diminishing returns property and states, informally, that adding an element to a smaller set increases the function value more than adding that element to a larger set. Submodular functions naturally model notions of coverage and diversity in applications, and therefore, a number of machine learning problems can be modeled as forms of submodular optimization (Kempe et al., 2003; Krause and Guestrin, 2005; Narasimhan and Bilmes, 2004; Iyer et al., 2013; Lin and Bilmes, 2012; Lin and Bilmes, 2010). In this paper, we investigate structured prediction methods for learn-

[4] http://en.wikipedia.org/wiki/Help:Categories

[5] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation

[6] http://en.wikipedia.org/wiki/Apple_(disambiguation)

ing weighted mixtures of submodular functions to summarize topics for a collection of objects using DAG-structured topic hierarchies. Throughout this paper we use the terms "topic" and "category" interchangeably.

1.1 Related Work

To the best of our knowledge, the specific problem we consider here is new. Previous work on identifying topics can be broadly categorized into one of the following types: a) cluster the objects and then identify names for the clusters; or b) dynamically identify topics (including hierarchical) for a set of objects. LDA (Blei et al., 2003) clusters the documents and simultaneously produces a set of topics into which the documents are clustered. In LDA, each document may be viewed as a mixture of various topics and the topic distribution is assumed to have a Dirichlet prior. LDA associates a group of high probability words to each identified topic. A name can be assigned to a topic by manually inspecting the words or using additional algorithms like (Mei et al., 2007; Maiya et al., 2013). LDA does not make use of existing topic hierarchies and correlation between topics. The Correlated Topic Model (Blei and Lafferty, 2006) induces a correlation structure between topics by using the logistic normal distribution instead of the Dirichlet. Another extension is the hierarchical LDA (Blei et al., 2004), where topics are joined together in a hierarchy by using the nested Chinese restaurant process. Nonparametric extensions of LDA include the Hierarchical Dirichlet Process (Teh et al., 2006) mixture model, which allows the number of topics to be unbounded and learnt from data and the Nested Chinese Restaurant Process which allows topics to be arranged in a hierarchy whose structure is learnt from data. In each of these approaches, unlike our proposed approach, an existing topic hierarchy is not used, nor is any additional object-topic information leveraged.

The pachinko allocation model (PAM)(Li and McCallum, 2006) captures arbitrary, nested, and possibly sparse correlations between topics using a DAG. The leaves of the DAG represent individual words in the vocabulary, while each interior node represents a correlation among its children, which may be words or other interior nodes (topics). PAM learns the probability distributions of words in a topic, subtopics in a topic, and topics in a document. We cannot, however, generate a subset of topics from a large existing topic DAG that can act as summary topics, using PAM.

HSLDA (Perotte et al., 2011) introduces a hierarchically supervised LDA model to infer hierarchical labels for a document. It assumes an existing label hierarchy in the form of a tree. The model infers one or more labels such that, if a label l is inferred as relevant to a document, then all the labels from l to the root of the tree are also inferred as relevant to the document. Our approach differs from HSLDA since: (1) we use the label hierarchy to infer a set of labels for a *group* of documents; (2) we do not enforce the label hierarchy to be a tree as it can be a DAG; and (3) generalizing HSLDA to use a DAG structured hierarchy and infer labels for a *group* of documents (e.g., combining into one big document) also may not help in solving our problem. HSLDA will apply *all* the relevant labels to the documents as per the classifier that it learns for every label. Moreover, the "root" label is always applied and it is very likely that many labels near the top level of the label hierarchy are also classified as relevant to the group of documents.

Wei and James (Bi and Kwok, 2011) present a hierarchical multi-label classification algorithm that can be used on both tree and DAG structured hierarchies. They formulate a search for the optimal consistent multi-label as the finding of the best subgraph in a tree/DAG. In our approach, we assume, individual documents are already associated with one or more topics and we find a consistent label set for a group of documents using the DAG structured topic hierarchy.

Medelyan et al. (Medelyan et al., 2008) and Ferragina et al. (Ferragina and Scaiella, 2010) detect topics for a document using Wikipedia article names and category names as the topic vocabulary. These systems are able to extract signals from a text document and identify Wikipedia articles and/or categories that optimally match the document and assign those article/category names as topics for the document. When run on a large collection of documents, these approaches generate enormous numbers of topics, a problem our proposed approach addresses.

1.2 Our Contributions

While most prior work discussed above focuses on the underlying set of documents, (*e.g.*, by clustering documents), we focus directly on the topics. In particular, we formulate the problem as subset selection on the set of topics within a DAG while simultaneously considering the documents to be categorized. Our method can scale to the colossal size of the DAG (1 million topics and 3 million correlation links between topics in Wikipedia). Moreover, our approach can

naturally incorporate outputs from many of the aforementioned algorithms. Our approach is based on submodular maximization and mixture learning, which has been successfully used in applications such as document summarization (Lin, 2012) and image summarization (Tschiatschek et al., 2014), but has never been applied to topic identification tasks or, more generally, DAG summarization.

We introduce a family of submodular functions to identify an appropriate set of topics from a DAG structured hierarchy of topics for a group of documents. We characterize this topic appropriateness through a set of desirable properties such as coverage, diversity, specificity, clarity, and relevance. Each of the submodular function components we consider are monotone, thereby ensuring a near optimal performance obtainable via a simple greedy algorithm for optimization.[7]. We also show how our technique naturally embodies outputs of other algorithms such as LDA, clustering, and classifications. Finally, we utilize a large margin formulation for learning mixtures of these submodular functions, and show how we can optimally learn them from training data.

Our approach demonstrates how to utilize the features collectively in the document space and the topic space to infer a set of topics. From an empirical perspective, we introduce and evaluate our approach on a dataset of around 8000 disambiguations that was extracted from Wikipedia and subsequently cleaned using the methods described in the experimentation section. We show that our learning framework outperforms many of the baselines, and is practical enough to be used on large corpora.

2 Problem Formulation

Let $G\,(V, E)$ be the DAG structured topic hierarchy with V topics. These topics are observed to have a parent child (*isa*) relationship forming a DAG. Let D be the set of documents that are associated with one or more of these topics. The middle portion of Figure 1 depicts a topic hierarchy with associated documents. The association links between the documents and topics can be hard or soft. In case of a hard link, a document is attached to a set of topics. Examples include multi-labeled documents. In case of a soft link, a document is associated with a topic with some degree of confidence (or probability). Furthermore, if a document is attached to a topic t, we assume that all the ancestor topics of t are also relevant for that document. This assumption has been employed in earlier works (Blei et al., 2004; Bi and Kwok, 2011; Rousu et al., 2006) as well. Given a budget of K, our objective is to choose a set of K topics from V, which best describe the documents in D. The notion of best describing topics is characterized through a set of desirable properties - coverage, diversity, specificity, clarity, relevance and fidelity - that K topics have to satisfy. The submodular functions that we introduce in the next section ensure these properties are satisfied. Formally, we solve the following discrete optimization problem:

$$S^* \in \underset{S \subseteq V : |S| \leq K}{\operatorname{argmax}} \sum_i w_i f_i(S) \qquad (1)$$

where, f_i are monotone submodular mixture components and $w_i \geq 0$ are the weights associated with those mixture components. Set S^* is the summary topics scored best.

It is easy to find massive (i.e., size in the order of million) DAG structured topic hierarchies in practice. Wikipedia's category hierarchy consists of more than 1M categories (topics) arranged hierarchically. In fact, they form a cyclic graph (Zesch and Gurevych, 2007). However, we can convert it to a DAG by eliminating the cycles as described in the supplementary material. YAGO (Suchanek et al., 2007) and Freebase (Bollacker et al., 2008) are other instances of massive topic hierarchies. The association of the documents with the existing topic hierarchy is also well studied. Systems such as WikipediaMiner (Milne, 2009), TAGME (Ferragina and Scaiella, 2010) and several annotation systems such as (Dill et al., 2003; Mihalcea and Csomai, 2007; Bunescu and Pasca, 2006) attach topics from Wikipedia (and other catalogs) to the documents by establishing the hard or soft links mentioned above.

Our goal is the following: Given a (ground set) collection V of topics organized in a pre-existing hierarchical DAG structure, and a collection D of documents, chose a size $K \in \mathbb{Z}_+$ representative subset of topics. Our approach is distinct from earlier work (e.g., (Kanungo et al., 2002; Blei et al., 2003)) where typically only a set of documents is classified and categorized in some way. We next provide a few definitions needed later in the paper.

Definition 1: Transitive Cover Γ): A topic t is said to cover a set of documents $\Gamma(t)$, called the transitive cover of the topic t, if for all documents $i \in \Gamma(t)$, either i is associated directly with topic t or with any of the descendant topics of t in the topic DAG. A natural extension of this definition to a set of topics T is defined as $\Gamma(T) = \cup_{t \in T} \Gamma(t)$.

[7] A simple greedy algorithm (Nemhauser et al., 1978) obtains a $1 - 1/e$ approximation guarantee for monotone submodular function maximization

Definition 2: Truncated Transitive Cover (Γ^α): This is a transitive cover of topic t, but with the limitation that the path length between a document and the topic t is not more than α. Hence, $|\Gamma^\alpha(t)| \leq |\Gamma(t)|$.

While our problem is closely related to clustering approaches, which consider the set of documents directly, there are some crucial differences. In particular, we focus on producing a clustering of documents where clusters are encouraged to honor a pre-defined DAG structured topic hierarchy. Existing agglomerative clustering algorithms focusing on the coverage of documents may not produce the desired clustering. To understand this, consider six documents d1, d2 ... d6 to be grouped into three clusters. There may be multiple ways to do this depending upon multiple aggregation paths present in the topic DAG: ((d1, d2), (d3, d4), (d5, d6)) or ((d1, d2, d3), (d4, d5), (d6)) or ((d1, d2, d3, d4), (d5), (d6)) or something else. Hence, we need more stringent measures to prefer one clustering over the others. Our work addresses this with a variety of quality criteria (coverage, diversity, specificity, clarity, relevance and fidelity, which are explained later in this paper) that are organically derived from well established submodular functions. And, most importantly, we learn the right mixture of these qualities to be enforced from the data itself. Furthermore, our approach also generalizes these clustering approaches, since one of the components in our mixture of submodular functions is defined via these unsupervised approaches, and maps a given clustering to a set of topics in the hierarchy.

3 Submodular Components and Learning

Summarization is the task of extracting information from a source that is both small in size but still representative. Our problem is different from traditional summarization tasks since we have an underlying DAG as a topic hierarchy that we wish to summarize in response to a subset of documents. Thus, a critical part of our problem is to take the graph structure into account while creating the summaries. Below, we identify properties we wish our summaries to posses.

Coverage: A summary set of topics should cover most of the documents. A document is said to be covered by a topic if there exists a path from the topic, going through intermediary descendant topics, to the document, i.e., the document is within the transitive cover of the topic.

Diversity: Summaries should be as diverse as possible, i.e., each summary topic should cover a unique set of documents. When a document is covered by more than one topic, that document is redundantly covered, e.g., "Finance" and "Banking" would be unlikely members of the same summary.

Summary qualities also involve "quality" notions, including:

Specificity/Clarity/Relevance/Coherence: These quality measures help us choose a set of topics that are neither too abstract nor overly specific. They ensure that the topics are clear and relevant to the documents that they represent. When additional information such as clustering (from LDA or other sources) and tagging (manual) documents is available, these quality criteria encourage the chosen topics to show resemblance (coherence) to those clustering/tagging in terms of transitive cover of documents they produce.

In the below, we define a variety of submodular functions that capture the above properties, and we then describe a large margin learning framework for learning convex mixtures of such components.

3.1 Submodular Components

3.1.1 Coverage Based Functions

Coverage components capture "coverage" of a set of documents.

Weighted Set Cover Function: Given a set of categories, $S \subseteq V$, define $\Gamma(S)$ as the set of documents covered — for each topic $s \in S$, $\Gamma(s) \subseteq D$ represents the documents covered by topic s and $\Gamma(S) = \cup_{s \in S} \Gamma(s)$. The weighted set cover function, defined as $f(S) = \sum_{d \in \Gamma(S)} w_d = w(\Gamma(S))$, assigns weights to the documents based on their relative importance (e.g., in Wikipedia disambiguation, the different documents could be ranked based on their priority).

Feature-based Functions: This class of function represents coverage in feature space. Given a set of categories $S \subseteq V$, and a set of features U, define $m_u(S)$ as the score associated with the set of categories S for feature $u \in U$. The feature set could represent, for example, the documents, in which case $m_u(S)$ represents the number of times document u is covered by the set S. U could also represent more complicated features. For example, in the context of Wikipedia disambiguation, U could represent TFIDF features over the documents. Feature based are then defined as $f(S) = \sum_{u \in U} \psi(m_u(S))$, where ψ is a concave (e.g., the square root) function. This function class has been successfully used in several applications (Kirchhoff and Bilmes, 2014; Wei et al., 2014a; Wei et al., 2014b).

3.1.2 Similarity based Functions

Similarity functions are defined through a similarity matrix $\mathbf{S} = \{s_{ij}\}_{i,j \in V}$. Given categories $i, j \in V$, similarity s_{ij} in our case can be defined as $s_{ij} = |\Gamma(i) \cap \Gamma(j)|$, i.e the number of documents commonly covered by both i and j.

Facility Location: The facility location function, defined as $f(S) = \sum_{i \in V} \max_{j \in S} s_{ij}$, is a natural model for k-medoids and exemplar based clustering, and has been used in several summarization problems (Tschiatschek et al., 2014; Wei et al., 2014a).

Penalty based diversity: A similarity matrix may be used to express a form of coverage of a set S but that is then penalized with a redundancy term, as in the following difference: $f(S) = \sum_{i \in V, j \in S} s_{ij} - \lambda \sum_{i \in S} \sum_{j \in S,} s_{i,j}$ (Lin and Bilmes, 2011)). Here $\lambda \in [0, 1]$. This function is submodular, but is not in general monotone, and has been used in document summarization (Lin and Bilmes, 2011), as a dispersion function (Borodin et al., 2012), and in image summarization (Tschiatschek et al., 2014).

3.1.3 Quality Control (QC) Functions

QC functions ensure a quality criteria is met by a set S of topics. We define the quality score of the set S as $F_q(S) = \sum_{s \in S} f_q(s)$, where $f_q(s)$ is the quality score of topic s for quality q. Therefore, $F_q(S)$ is a modular function in S. We investigate three types of quality control functions: Topic Specificity, Topic Clarity, and Topic Relevance.

Topic Specificity: The farther a topic is from the root of the DAG, the more specific it becomes. Topics higher up in the hierarchy are abstract and less specific. We therefore prefer topics low in the DAG, but lower topics also have less coverage. We define $f_{\text{specificity}}(s) = s_h$ where s_h is the height of topic s in the DAG. The root topic has height zero and the "height" increases as we move down the DAG in Figure 1.

Topic Clarity: Topic clarity is the fraction of descendant topics that cover one or more documents. If a topic has many descendant topics that do not cover any documents, it has less clarity. Formally, $f_{\text{clarity}}(s) = \frac{\sum_{t \in \text{descendants}(s)} [\![\Gamma(t) > 0]\!]}{|\text{descendants}(s)|}$, where $[\![.]\!]$ is the indicator function.

Topic Relevance: A topic is considered to be better related to a document if the number of hops needed to reach the document from that topic is lower. Given any set $A \subseteq D$ of document, and any topic $s \in V$, we can define $f_{\text{relevance}}(s|A) = \text{argmin}_\alpha \{\alpha : A \subseteq \Gamma^\alpha(s)\}$.

QC Functions As Barrier Modular Mixtures: We introduce a modular function for every QC function as follows

$$f^\alpha_{\text{specificity}}(s) = \begin{cases} 1 & \text{if the height of topic } s \text{ is at least } \alpha \\ 0 & \text{otherwise} \end{cases}$$

for every possible value of α. This creates a submodular mixture with as many components as the number of possible values of α. In our experiments with Wikipedia, we had α varying from 1 to 120 stepping by 1, adding 120 modular mixture components. Similarly, we define,

$$f^\beta_{\text{clarity}}(s) = \begin{cases} 1 & \text{if the clarity of topic } s \text{ is at least } \beta \\ 0 & \text{otherwise} \end{cases}$$

for every possible (discretized to make it countably finite) value of β. And,

$f^\gamma_{\text{relevance}}(s) = f_{\text{cov}}(s|\Gamma^\gamma(s))$, where $f_{\text{cov}}(.)$ is the coverage submodular function and $s|X$ indicates coverage of a topic s over a set of documents X. All these functions (modular and submodular terms) are added as mixture components in our learning framework to learn suitable weights for them. We then use these weights in our inference procedure to obtain a subset of topics as described in 3.2. We show from our experiments that this approach performs better than all other approaches and baselines.

3.1.4 Fidelity Functions

A function representing the fidelity of a set S to another reference set R is one that gets a large value when the set S represents the set R. Such a function scores inferred topics high when it resembles a reference set of topics and/or item clusters. The reference set in this case can be produced from other algorithms such as k-means, LDA and its variants or from a manually tagged corpus. Next we describe one such fidelity function.

Topic Coherence: This function scores a set of topics S high when the transitive cover (Definition 1) produced by the topics in S resembles the clusters of documents produced by an external source (k-means, LDA or manual). Given an external source that clusters the documents, producing T clusters $L_1, L_2, ..., L_T$ (for T topics), topic coherence is defined as: $f(S) = \sum_{t \in T} \max_{k \in S} w_{k,t}$ where $w_{k,t} = \text{harmonic\_mean}(w^p_{k,t}, w^r_{k,t})$ and $w^p_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|\Gamma(k)|}$ and $w^r_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|L_t|}$. Note that, $w^p_{k,t} \geq 0$ and $w^r_{k,t} \geq 0$ are the precision on recall of the resemblance and $w_{k,t}$ is the F1 measure. If the transitive cover of topics in S resembles the reference clusters L_t exactly, we attain maximum coherence (or fidelity). As the resemblance diminishes, the score decreases. The above function $f(S)$ is monotone submodular.

3.1.5 Mixture of Submodular Components:

Given the different classes of submodular functions above, we construct our submodular scoring functions $F_w(\cdot)$ as a convex combinations of these different submodular functions $f_1, f_2, \ldots, f_m$, above. In other words,

$$F_w(S) = \sum_{i=1}^{m} w_i f_i(S), \tag{2}$$

where $w = (w_1, \ldots, w_m)$, $w_i \geq 0, \sum_i w_i = 1$. The components f_i are submodular and assumed to be normalized: i.e., $f_i(\emptyset) = 0$, and $f_i(V) = 1$ for monotone functions and $\max_{A \subseteq V} f_i(A) \leq 1$ for non-monotone functions. A simple way to normalize a monotone submodular function is to define the component as $f_i(S)/f_i(V)$. This ensures that the components are *compatible* with each other. Obviously, the merit of the scoring function $F_w(\cdot)$ depends on the selection of the components.

3.2 Large Margin Learning

We optimize the weights w of the scoring function $F_w(\cdot)$ in a large-margin structured prediction framework. In this setting, we assume we have training data in the form of pairs of a set of documents, and a human generated summary as a set of topics. For example, in the case of Wikipedia disambiguation, we use the human generated disambiguation pages as the ground truth summary. We represent the set of ground-truth summaries as $\mathcal{S} = \{S_1, S_2, \cdots, S_N\}$. In large margin training, the weights are optimized such that ground-truth summaries $\mathcal{S}$ are separated from competitor summaries by a loss-dependent margin:

$$F_w(S) \geq F_w(S') + \mathcal{L}(S'), \quad \forall S \in \mathcal{S}, S' \in \mathcal{Y} \setminus \mathcal{S}, \tag{3}$$

where $\mathcal{L}(\cdot)$ is the loss function, and where $\mathcal{Y}$ is a structured output space (for example $\mathcal{Y}$ is the set of summaries that satisfy a certain budget B, i.e., $\mathcal{Y} = \{S' \subseteq V : |S'| \leq B\}$). We assume the loss to be normalized, $0 \leq \mathcal{L}(S') \leq 1, \forall S' \subseteq V$, to ensure that mixture and loss are calibrated. Equation (3) can be stated as $F_w(S) \geq \max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right], \forall S \in \mathcal{S}$ which is called *loss-augmented inference*. We introduce slack variables and minimize the regularized sum of slacks (Lin and Bilmes, 2012):

$$\min_{w \geq 0, \|w\|_1 = 1} \quad \sum_{S \in \mathcal{S}} \left[\max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right] - F_w(S)\right] + \frac{\lambda}{2}\|w\|_2^2, \tag{4}$$

where the non-negative orthant constraint, $w \geq 0$, ensures that the final mixture is submodular. Note a 2-norm regularizer is used on top of a 1-norm constraint $\|w\|_1 = 1$ which we interpret as a prior to encourage higher entropy, and thus more diverse mixture distributions. Tractability depends on the choice of the loss function. The parameters w are learnt using stochastic gradient descent as in (Tschiatschek et al., 2014).

3.3 Loss Functions

A natural choice of loss functions for our case can be derived from cluster evaluation metrics. Every inferred topic s induces a subset of documents, namely the transitive cover $\Gamma(s)$ of s. We compare these clusters with the clusters induced from the true topics in the training set and compute the loss.

In this paper, we use the Jaccard Index (JI) as a loss function. Let S be the inferred topics and T be the true topics. The Jaccard loss is defined as $\mathcal{L}_{\text{jaccard}}(S,T) = 1 - \frac{1}{k}\sum_{s \in S} \max_{t \in T} \frac{|\Gamma(s) \cap \Gamma(t)|}{|\Gamma(s) \cup \Gamma(t)|}$, where $k = |S| = |T|$ is the number of topics. When the clustering produced by the inferred and the true topics are similar, Jaccard loss is 0. When they are completely dissimilar, the loss is maximum, i.e., 1. Jaccard loss is a modular function.

3.4 Inference Algorithm: Greedy

Having learnt the weights for the mixture components, the resulting function $F_w(S) = \sum_{i=1}^{m} w_i f_i(S)$ is a submodular function. In the case when the individual components are themselves monotone (all our functions in fact are), $F_w(S)$ can be optimized by the accelerated greedy algorithm (Minoux, 1978). Thanks to submodularity, we can obtain near optimal solutions very efficiently. In case the functions are all monotone submodular, we can guarantee that the solution is within $1 - 1/e$ factor from the optimal solution.

4 Experimental Results

To validate our approach, we make use of Wikipedia category structure as a topic DAG and apply our technique to the task of automatic generation of Wikipedia disambiguation pages. We pre-processed the category graph to eliminate the cycles in order to make it a DAG. Each Wikipedia disambiguation page is manually created by Wikipedia editors by grouping a collection of Wikipedia articles into several groups. Each group is then assigned a name, which serves as a topic for the group. Typically, a disambiguation page segregates around 20-30 articles into 5-6 groups. Our goal is to measure how accurately we can recreate the groups for a disambiguation page and label them, given only the collection of articles mentioned in that disambiguation page (when actual groupings and labels are hidden).

4.1 Datasets

We parsed the contents of Wikipedia disambiguation pages and extracted disambiguation page names, article groups and group names. We collected about 8000 disambiguation pages that had at least four groups on them. Wikipedia category structure is used as the topic DAG. We eliminated few administrative categories such as "Hidden Categories", "Articles needing cleanup", and the like. The final DAG had about 1M topics and 3M links.

4.2 Evaluation Metrics

Every group of articles on the Wikipedia disambiguation page is assigned a name by the editors. Unfortunately, these names may not correspond to the Wikipedia category (topic) names. For example, one of the groups on the "Matrix" disambiguation page has a name "Business and government" and there is no Wikipedia category by that name. However, the group names generated by our (and baseline) method are from the Wikipedia categories (which forms our topic DAG). In addition, there can be multiple relevant names for a group. For example, a group on a disambiguation page may be called "Calculus", but an algorithm may rightly generate "Vector Calculus". Hence we cannot evaluate the accuracy of an algorithm just by matching the generated group names to those on the disambiguation page. To alleviate this problem, we adopt cluster-based evaluation metrics. We treat every group of articles generated by an algorithm under a topic for a disambiguation page as a cluster of articles. These are considered as *inferred* clusters for a disambiguation page. We compare them against the actual grouping of articles on the Wikipedia disambiguation page by treating those groups as *true* clusters. We can now adopt Jaccard Index, F1-measure, and NMI (Normalized Mutual Information) based cluster evaluation metrics described in (Manning et al., 2008). For each disambiguation page in the test set, we compute every metric score and then average it over all the disambiguation pages.

4.3 Methods Compared

We validated our approach by comparing against several baselines described below. We also compared two variations of our approach as described next. For each of these cases (baselines and two variations) we generated and compared the metrics (Jaccard Index, F1-measure and NMI) as described in the previous section.

KM_{docs}: K-Means algorithm run on articles as TF-IDF vectors of words. The number of clusters K is set to the number of true clusters on the Wikipedia disambiguation page.

$\text{KMed}_{\text{docs}}$: K-Medoids algorithm with articles as TF-IDF vectors of words. The number of clusters are set as in KM_{docs}.

$\text{KMed}_{\text{topics}}$: K-Medoids run on topics as TF-IDF vectors of words. The words for each topic is taken from the articles that are in the transitive cover of the topic.

LDA_{docs}: LDA algorithm with the number of topics set to the number of true clusters on the Wikipedia disambiguation page. Each article is then grouped under the highest probability topic.

SMML_{cov}: This is the submodular mixture learning case explained in section 3.1.5. Here we consider a mixture of all the submodular functions governing coverage, diversity, fidelity and QC functions. However, we exclude the similarity based functions described in section 3.1.2. Coverage based functions have a time complexity of $O(n)$ whereas similarity based functions are $O(n^2)$. By excluding similarity based functions, we can compare the quality of the results with and without $O(n^2)$ functions. We learn the mixture weights from the training set and use them during inference on the test set to subset K topics through the submodular maximization (Equation 1).

$\text{SMML}_{\text{cov+sim}}$: This case is similar to SMML_{cov} except that, we include similarity based submodular mixture components. This makes the inference time complexity $O(n^2)$.

We do not compare against HSLDA, PAM and few other techniques cited in the related work sections because they do not produce a subset of K summary topics — these are not directly comparable with our work.

4.4 Evaluation Results

We show that the submodular mixture learning and maximization approaches, i.e., SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other approaches in various metrics. In all these experiments, we performed 5 fold cross validation to learn the parameters from 80% of the disambiguation pages and evaluated on the rest of the 20%, in each fold.

In Figure 2a we summarize the results of the comparison of the methods mentioned above on Jaccard Index, F1 measure and NMI. Our proposed techniques SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other techniques consistently.

In Figures 2b and 2c we measure the number of test instances (i.e., disambiguation queries) in which each of the algorithms dominate (win) in evaluation metrics. In 60% of the disambiguation queries, SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ approaches

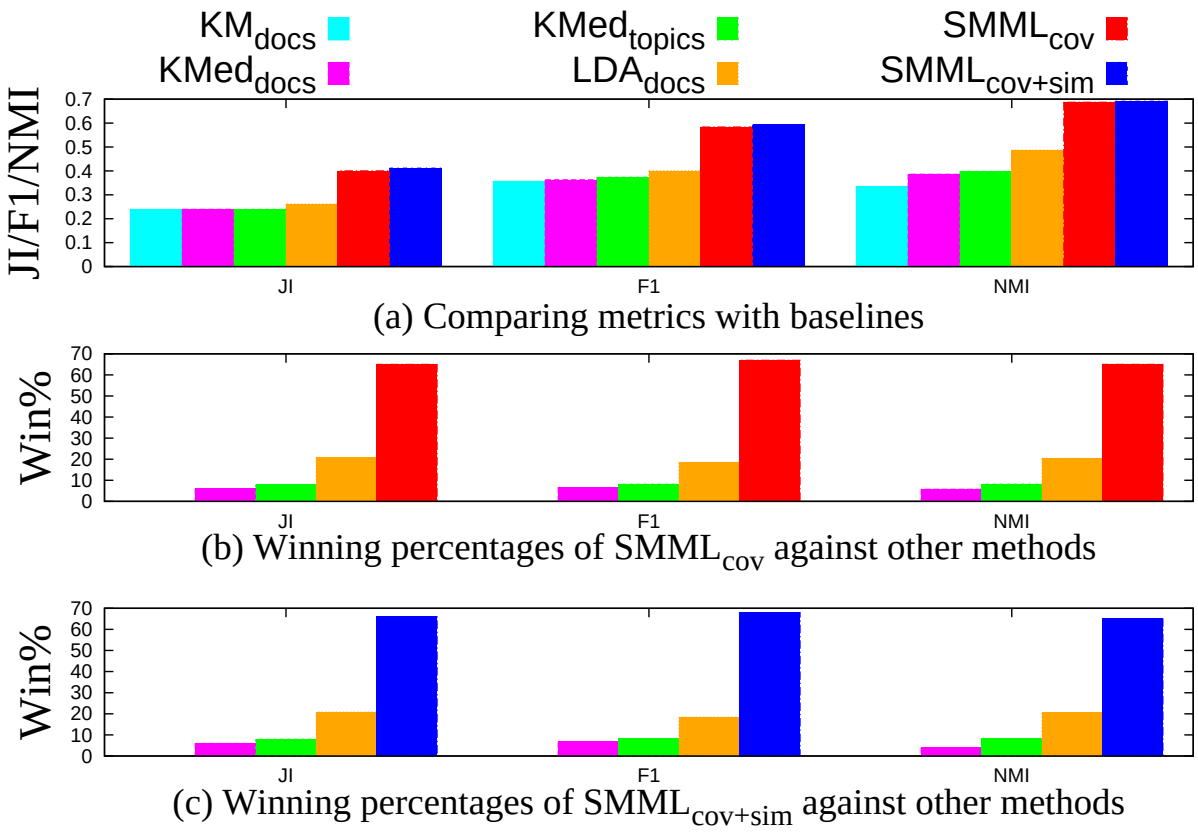

Figure 2: Comparison of techniques

produce higher JI, F1 and NMI than all other methods. This indicates that the clusters of articles produced by our technique resembles the clusters of articles present on the disambiguation page better than other techniques.

From Figures 2b and 2c it is clear that $O(n)$ time complexity based submodular mixture functions (SMML_{cov}) perform on par with $O(n^2)$ based functions ($\text{SMML}_{\text{cov+sim}}$), but at a greatly reduced execution time, demonstrating the sufficiency of $O(n)$ functions for our task. On the average, for each disambiguation query, SMML_{cov} took around 40 seconds (over 1M topics and 3M edges DAG) to infer the topics, whereas $\text{SMML}_{\text{cov+sim}}$ took around 35 minutes. Both these experiments were carried on a machine with 32 GB RAM, Eight-Core AMD Opteron(tm) Processor 2427.

5 Conclusions

We investigated a problem of summarizing topics over a massive topic DAG such that the summary set of topics produced represents the objects in the collection. This representation is characterized through various classes of submodular (and monotone) functions that captured coverage, similarity, diversity, specificity, clarity, relevance and fidelity of the topics. Currently we assume that the number of topics K is given as an input to our algorithm. It would be an interesting future problem to estimate the value of K automatically in our setting. As future work, we also plan to extend our techniques to produce a hierarchical summary of topics and scale it across heterogeneous collection of objects (from different domains) to bring all of them under the same topic DAG and investigate interesting cases thereon.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1162606, and by a Google, a Microsoft, and an Intel research award. Rishabh Iyer acknowledges support from the Microsoft Research Ph.D Fellowship.

References

Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. 2006. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, April.

W. Bi and J. T. Kwok. 2011. Multi-label classification on tree-and DAG-structured hierarchies. In *ICML*. ICML.

David M. Blei and John D. Lafferty. 2006. Correlated topic models. In *In Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. MIT Press.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, page 2003. MIT Press.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of Principles of Database Systems*, pages 155–166. ACM.

Razvan Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06), Trento, Italy*, pages 9–16, April.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. 2003. Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 178–186, New York, NY, USA. ACM.

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA.

R. Iyer, S. Jegelka, and J. Bilmes. 2013. Fast semidifferential-based submodular function optimization. ICML.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, Angela Y. Wu, Senior Member, and Senior Member. 2002. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:881–892.

D. Kempe, J. Kleinberg, and E. Tardos. 2003. Maximizing the spread of influence through a social network. In *SIGKDD*.

Katrin Kirchhoff and Jeff Bilmes. 2014. Submodularity for data selection in machine translation. October.

A. Krause and C. Guestrin. 2005. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of Uncertainity in Artificial Intelligence*. UAI.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 577–584, New York, NY, USA. ACM.

H. Lin and J. Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL*.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *The 49th Meeting of the Assoc. for Comp. Ling. Human Lang. Technologies (ACL/HLT-2011)*, Portland, OR, June.

H. Lin and J. Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, page 479490.

Hui Lin. 2012. *Submodularity in Natural Language Processing: Algorithms and Applications*. Ph.D. thesis, University of Washington, Dept. of EE.

Min ling Zhang and Zhi hua Zhou. 2007. Ml-knn: A lazy learning approach to multi-label learning. *PATTERN RECOGNITION*, 40:2007.

Arun S. Maiya, John P. Thompson, Francisco Loaiza-Lemos, and Robert M. Rolfe. 2013. Exploratory analysis of highly heterogeneous document collections. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1375–1383, New York, NY, USA. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Olena Medelyan, Ian H. Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the Wikipedia and AI workshop at AAAI-08*. AAAI.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 490–499, New York, NY, USA. ACM.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA. ACM.

David Milne. 2009. An open-source toolkit for mining wikipedia. In *In Proc. New Zealand Computer Science Research Student Conf*, page 2009.

Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, chapter 27, pages 234–243. Springer Berlin Heidelberg, Berlin/Heidelberg.

Mukund Narasimhan and Jeff Bilmes. 2004. PAC-learning bounded tree-width graphical models. In *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference (UAI-2004)*. Morgan Kaufmann Publishers, July.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294.

Adler J. Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. 2011. Hierarchically supervised latent dirichlet allocation. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *NIPS*, pages 2609–2617.

Juho Rousu, Craig Saunders, Sndor Szedmk, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626.

Jr. Silla, CarlosN. and AlexA. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Sebastian Tschiatschek, Rishabh Iyer, Hoachen Wei, and Jeff Bilmes. 2014. Learning Mixtures of Submodular Functions for Image Collection Summarization. In *Neural Information Processing Systems (NIPS)*.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer US.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2014a. Fast multi-stage submodular maximization. In *ICML*.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, Chris Bartels, and Jeff Bilmes. 2014b. Submodular subset selection for large-scale speech training data. *Proceedings of ICASSP, Florence, Italy*.

Torsten Zesch and Iryna Gurevych. 2007. Analysis of the wikipedia category graph for nlp applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, pages 1–8, Rochester, April. Association for Computational Linguistics.

Summarization of Multi-Document Topic Hierarchies using Submodular Mixtures

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, 40076, India
bairi@cse.iitb.ac.in

Rishabh Iyer
University of Washington
Seattle, WA-98175, USA
rkiyer@u.washington.edu

Ganesh Ramakrishnan
IIT Bombay
Mumbai, 40076, India
ganesh@cse.iitb.ac.in

Jeff Bilmes
University of Washington
Seattle, WA-98175, USA
bilmes@uw.edu

Abstract

We study the problem of summarizing DAG-structured topic hierarchies over a given set of documents. Example applications include automatically generating Wikipedia disambiguation pages for a set of articles, and generating candidate multi-labels for preparing machine learning datasets (e.g., for text classification, functional genomics, and image classification). Unlike previous work, which focuses on clustering the set of documents using the topic hierarchy as features, we directly pose the problem as a submodular optimization problem on a topic hierarchy using the documents as features. Desirable properties of the chosen topics include document coverage, specificity, topic diversity, and topic homogeneity, each of which, we show, is naturally modeled by a submodular function. Other information, provided say by unsupervised approaches such as LDA and its variants, can also be utilized by defining a submodular function that expresses coherence between the chosen topics and this information. We use a large-margin framework to learn convex mixtures over the set of submodular components. We empirically evaluate our method on the problem of automatically generating Wikipedia disambiguation pages using human generated clusterings as ground truth. We find that our framework improves upon several baselines according to a variety of standard evaluation metrics including the Jaccard Index, F1 score and NMI, and moreover, can be scaled to extremely large scale problems.

1 Introduction

Several real world machine learning applications involve hierarchy based categorization of topics for a set of objects. Objects could be, e.g., a set of documents for text classification, a set of genes in functional genomics, or a set of images in computer vision. One can often define a natural topic hierarchy to categorize these objects. For example, in text and image classification problems, each document or image is assigned a hierarchy of labels — a baseball page is assigned the labels "baseball" and "sports." Moreover, many of these applications, naturally have an existing topic hierarchy generated on the entire set of objects (Rousu et al., 2006; Barutcuoglu et al., 2006; ling Zhang and hua Zhou, 2007; Silla and Freitas, 2011; Tsoumakas et al., 2010).

Given a DAG-structured topic hierarchy and a subset of objects, we investigate the problem of finding a subset of DAG-structured topics that are induced by that subset (of objects). This problem arises naturally in several real world applications. For example, consider the problem of identifying appropriate label sets for a collection of articles. Several existing text collection datasets such as 20 Newsgroup[1], Reuters-21578[2] work with a predefined set of topics. We observe that these topic names are highly abstract[3] for the articles categorized under them. On the other hand, techniques proposed by systems such as Wikipedia Miner (Milne, 2009) and TAGME (Ferragina and Scaiella, 2010) generate several labels for each article in the dataset that are highly specific to the article. Collating all labels from all articles to create a label

[1] http://qwone.com/~jason/20Newsgroups/

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[3] Topic *Concept* is more abstract than the topic *Science* which is more abstract than the topic*Chemistry*

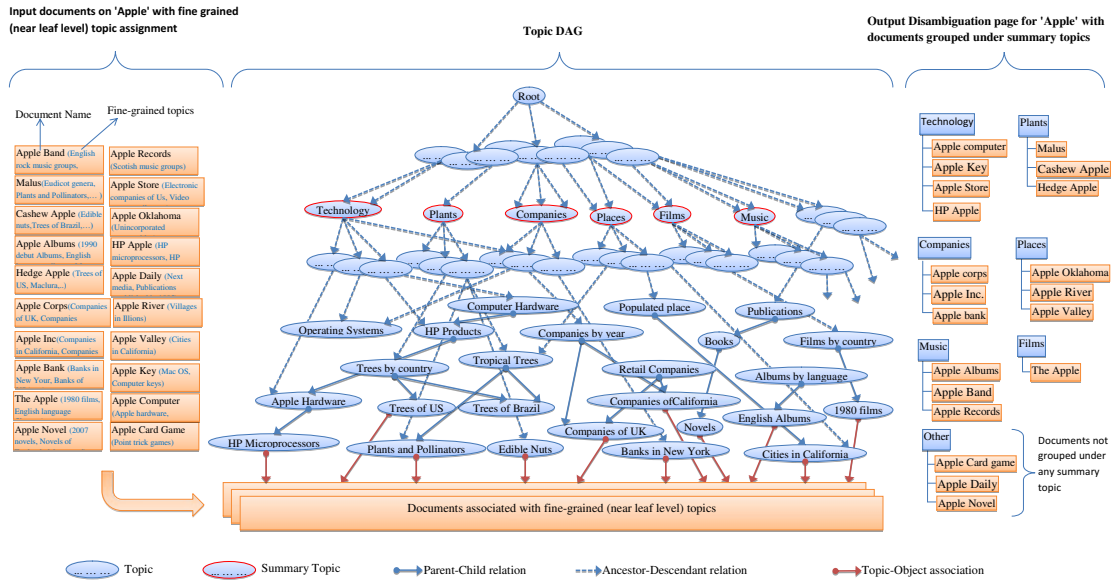

Figure 1: Topic Summarization overview. On the left, we show many documents related to Apple. In the middle, a Wikipedia category hierarchy shown as a topic DAG, links these documents at the leaf level. On the right, we show the output of our summarization process, which creates a set of summary topics (Plants, Technology, Companies, Films, Music and Places in this example) with the input documents classified under them.

set for the dataset can result in a large number of labels and become unmanageable. Our proposed techniques can summarize such large sets of labels into a smaller and more meaningful label sets using a DAG-structured topic hierarchy. This also holds for image classification problems and datasets like ImageNet (Deng et al., 2009). We use the term *summarize* to highlight the fact that the smaller label set semantically covers the larger label set. For example, the topics *Physics*, *Chemistry*, and *Mathematics* can be summarized into a topic *Science*.

A particularly important application of our work (and the one we use for our evaluations in Section 4) is the following: Given a collection of articles spanning different topics, but with similar titles, automatically generate a disambiguation page for those titles using the Wikipedia category hierarchy[4] as a topic DAG. Disambiguation pages[5] on Wikipedia are used to resolve conflicts in article titles that occur when a title is naturally associated with multiple articles on distinct topics. Each disambiguation page organizes articles into several groups, where the articles in each group pertain only to a specific topic. Disambiguations may be seen as paths in a hierarchy leading to different articles that arguably could have the same title. For example, the title *Apple*[6] can refer to a plant, a company, a film, a television show, a place, a technology, an album, a record label, and a newspaper daily. The problem then, is to organize the articles into multiple groups where each group contains articles of similar nature (topics) and has an appropriately discerned group heading. Figure 1 describes the topic summarization process for creation of the disambiguation page for "Apple".

All the above mentioned problems can be modeled as the problem of finding the most representative subset of topic nodes from a DAG-Structured topic hierarchy. We argue that many formulations of this problem are natural instances of submodular maximization, and provide a learning framework to create submodular mixtures to solve this problem. A set function $\mathrm{f}(.)$ is said to be submodular if for any element v and sets $A \subseteq B \subseteq V \setminus \{v\}$, where V represents the ground set of elements, $\mathrm{f}(A \cup \{v\}) - \mathrm{f}(A) \geq \mathrm{f}(B \cup \{v\}) - \mathrm{f}(B)$. This is called the diminishing returns property and states, informally, that adding an element to a smaller set increases the function value more than adding that element to a larger set. Submodular functions naturally model notions of coverage and diversity in applications, and therefore, a number of machine learning problems can be modeled as forms of submodular optimization (Kempe et al., 2003; Krause and Guestrin, 2005; Narasimhan and Bilmes, 2004; Iyer et al., 2013; Lin and Bilmes, 2012; Lin and Bilmes, 2010). In this paper, we investigate structured prediction methods for learn-

[4] http://en.wikipedia.org/wiki/Help:Categories

[5] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation

[6] http://en.wikipedia.org/wiki/Apple_(disambiguation)

ing weighted mixtures of submodular functions to summarize topics for a collection of objects using DAG-structured topic hierarchies. Throughout this paper we use the terms "topic" and "category" interchangeably.

1.1 Related Work

To the best of our knowledge, the specific problem we consider here is new. Previous work on identifying topics can be broadly categorized into one of the following types: a) cluster the objects and then identify names for the clusters; or b) dynamically identify topics (including hierarchical) for a set of objects. LDA (Blei et al., 2003) clusters the documents and simultaneously produces a set of topics into which the documents are clustered. In LDA, each document may be viewed as a mixture of various topics and the topic distribution is assumed to have a Dirichlet prior. LDA associates a group of high probability words to each identified topic. A name can be assigned to a topic by manually inspecting the words or using additional algorithms like (Mei et al., 2007; Maiya et al., 2013). LDA does not make use of existing topic hierarchies and correlation between topics. The Correlated Topic Model (Blei and Lafferty, 2006) induces a correlation structure between topics by using the logistic normal distribution instead of the Dirichlet. Another extension is the hierarchical LDA (Blei et al., 2004), where topics are joined together in a hierarchy by using the nested Chinese restaurant process. Nonparametric extensions of LDA include the Hierarchical Dirichlet Process (Teh et al., 2006) mixture model, which allows the number of topics to be unbounded and learnt from data and the Nested Chinese Restaurant Process which allows topics to be arranged in a hierarchy whose structure is learnt from data. In each of these approaches, unlike our proposed approach, an existing topic hierarchy is not used, nor is any additional object-topic information leveraged.

The pachinko allocation model (PAM)(Li and McCallum, 2006) captures arbitrary, nested, and possibly sparse correlations between topics using a DAG. The leaves of the DAG represent individual words in the vocabulary, while each interior node represents a correlation among its children, which may be words or other interior nodes (topics). PAM learns the probability distributions of words in a topic, subtopics in a topic, and topics in a document. We cannot, however, generate a subset of topics from a large existing topic DAG that can act as summary topics, using PAM.

HSLDA (Perotte et al., 2011) introduces a hierarchically supervised LDA model to infer hierarchical labels for a document. It assumes an existing label hierarchy in the form of a tree. The model infers one or more labels such that, if a label l is inferred as relevant to a document, then all the labels from l to the root of the tree are also inferred as relevant to the document. Our approach differs from HSLDA since: (1) we use the label hierarchy to infer a set of labels for a *group* of documents; (2) we do not enforce the label hierarchy to be a tree as it can be a DAG; and (3) generalizing HSLDA to use a DAG structured hierarchy and infer labels for a *group* of documents (e.g., combining into one big document) also may not help in solving our problem. HSLDA will apply *all* the relevant labels to the documents as per the classifier that it learns for every label. Moreover, the "root" label is always applied and it is very likely that many labels near the top level of the label hierarchy are also classified as relevant to the group of documents.

Wei and James (Bi and Kwok, 2011) present a hierarchical multi-label classification algorithm that can be used on both tree and DAG structured hierarchies. They formulate a search for the optimal consistent multi-label as the finding of the best subgraph in a tree/DAG. In our approach, we assume, individual documents are already associated with one or more topics and we find a consistent label set for a group of documents using the DAG structured topic hierarchy.

Medelyan et al. (Medelyan et al., 2008) and Ferragina et al. (Ferragina and Scaiella, 2010) detect topics for a document using Wikipedia article names and category names as the topic vocabulary. These systems are able to extract signals from a text document and identify Wikipedia articles and/or categories that optimally match the document and assign those article/category names as topics for the document. When run on a large collection of documents, these approaches generate enormous numbers of topics, a problem our proposed approach addresses.

1.2 Our Contributions

While most prior work discussed above focuses on the underlying set of documents, (*e.g.*, by clustering documents), we focus directly on the topics. In particular, we formulate the problem as subset selection on the set of topics within a DAG while simultaneously considering the documents to be categorized. Our method can scale to the colossal size of the DAG (1 million topics and 3 million correlation links between topics in Wikipedia). Moreover, our approach can

naturally incorporate outputs from many of the aforementioned algorithms. Our approach is based on submodular maximization and mixture learning, which has been successfully used in applications such as document summarization (Lin, 2012) and image summarization (Tschiatschek et al., 2014), but has never been applied to topic identification tasks or, more generally, DAG summarization.

We introduce a family of submodular functions to identify an appropriate set of topics from a DAG structured hierarchy of topics for a group of documents. We characterize this topic appropriateness through a set of desirable properties such as coverage, diversity, specificity, clarity, and relevance. Each of the submodular function components we consider are monotone, thereby ensuring a near optimal performance obtainable via a simple greedy algorithm for optimization.[7]. We also show how our technique naturally embodies outputs of other algorithms such as LDA, clustering, and classifications. Finally, we utilize a large margin formulation for learning mixtures of these submodular functions, and show how we can optimally learn them from training data.

Our approach demonstrates how to utilize the features collectively in the document space and the topic space to infer a set of topics. From an empirical perspective, we introduce and evaluate our approach on a dataset of around 8000 disambiguations that was extracted from Wikipedia and subsequently cleaned using the methods described in the experimentation section. We show that our learning framework outperforms many of the baselines, and is practical enough to be used on large corpora.

2 Problem Formulation

Let $G\,(V, E)$ be the DAG structured topic hierarchy with V topics. These topics are observed to have a parent child (*isa*) relationship forming a DAG. Let D be the set of documents that are associated with one or more of these topics. The middle portion of Figure 1 depicts a topic hierarchy with associated documents. The association links between the documents and topics can be hard or soft. In case of a hard link, a document is attached to a set of topics. Examples include multi-labeled documents. In case of a soft link, a document is associated with a topic with some degree of confidence (or probability). Furthermore, if a document is attached to a topic t, we assume that all the ancestor topics of t are also relevant for that document. This assumption has been employed in earlier works (Blei et al., 2004; Bi and Kwok, 2011; Rousu et al., 2006) as well. Given a budget of K, our objective is to choose a set of K topics from V, which best describe the documents in D. The notion of best describing topics is characterized through a set of desirable properties - coverage, diversity, specificity, clarity, relevance and fidelity - that K topics have to satisfy. The submodular functions that we introduce in the next section ensure these properties are satisfied. Formally, we solve the following discrete optimization problem:

$$S^* \in \underset{S \subseteq V : |S| \leq K}{\operatorname{argmax}} \sum_i w_i f_i(S) \qquad (1)$$

where, f_i are monotone submodular mixture components and $w_i \geq 0$ are the weights associated with those mixture components. Set S^* is the summary topics scored best.

It is easy to find massive (i.e., size in the order of million) DAG structured topic hierarchies in practice. Wikipedia's category hierarchy consists of more than 1M categories (topics) arranged hierarchically. In fact, they form a cyclic graph (Zesch and Gurevych, 2007). However, we can convert it to a DAG by eliminating the cycles as described in the supplementary material. YAGO (Suchanek et al., 2007) and Freebase (Bollacker et al., 2008) are other instances of massive topic hierarchies. The association of the documents with the existing topic hierarchy is also well studied. Systems such as WikipediaMiner (Milne, 2009), TAGME (Ferragina and Scaiella, 2010) and several annotation systems such as (Dill et al., 2003; Mihalcea and Csomai, 2007; Bunescu and Pasca, 2006) attach topics from Wikipedia (and other catalogs) to the documents by establishing the hard or soft links mentioned above.

Our goal is the following: Given a (ground set) collection V of topics organized in a pre-existing hierarchical DAG structure, and a collection D of documents, chose a size $K \in \mathbb{Z}_+$ representative subset of topics. Our approach is distinct from earlier work (e.g., (Kanungo et al., 2002; Blei et al., 2003)) where typically only a set of documents is classified and categorized in some way. We next provide a few definitions needed later in the paper.

Definition 1: Transitive Cover Γ): A topic t is said to cover a set of documents $\Gamma(t)$, called the transitive cover of the topic t, if for all documents $i \in \Gamma(t)$, either i is associated directly with topic t or with any of the descendant topics of t in the topic DAG. A natural extension of this definition to a set of topics T is defined as $\Gamma(T) = \cup_{t \in T} \Gamma(t)$.

[7] A simple greedy algorithm (Nemhauser et al., 1978) obtains a $1 - 1/e$ approximation guarantee for monotone submodular function maximization

Definition 2: Truncated Transitive Cover (Γ^α): This is a transitive cover of topic t, but with the limitation that the path length between a document and the topic t is not more than α. Hence, $|\Gamma^\alpha(t)| \leq |\Gamma(t)|$.

While our problem is closely related to clustering approaches, which consider the set of documents directly, there are some crucial differences. In particular, we focus on producing a clustering of documents where clusters are encouraged to honor a pre-defined DAG structured topic hierarchy. Existing agglomerative clustering algorithms focusing on the coverage of documents may not produce the desired clustering. To understand this, consider six documents d1, d2 ... d6 to be grouped into three clusters. There may be multiple ways to do this depending upon multiple aggregation paths present in the topic DAG: ((d1, d2), (d3, d4), (d5, d6)) or ((d1, d2, d3), (d4, d5), (d6)) or ((d1, d2, d3, d4), (d5), (d6)) or something else. Hence, we need more stringent measures to prefer one clustering over the others. Our work addresses this with a variety of quality criteria (coverage, diversity, specificity, clarity, relevance and fidelity, which are explained later in this paper) that are organically derived from well established submodular functions. And, most importantly, we learn the right mixture of these qualities to be enforced from the data itself. Furthermore, our approach also generalizes these clustering approaches, since one of the components in our mixture of submodular functions is defined via these unsupervised approaches, and maps a given clustering to a set of topics in the hierarchy.

3 Submodular Components and Learning

Summarization is the task of extracting information from a source that is both small in size but still representative. Our problem is different from traditional summarization tasks since we have an underlying DAG as a topic hierarchy that we wish to summarize in response to a subset of documents. Thus, a critical part of our problem is to take the graph structure into account while creating the summaries. Below, we identify properties we wish our summaries to posses.

Coverage: A summary set of topics should cover most of the documents. A document is said to be covered by a topic if there exists a path from the topic, going through intermediary descendant topics, to the document, i.e., the document is within the transitive cover of the topic.

Diversity: Summaries should be as diverse as possible, i.e., each summary topic should cover a unique set of documents. When a document is covered by more than one topic, that document is redundantly covered, e.g., "Finance" and "Banking" would be unlikely members of the same summary.

Summary qualities also involve "quality" notions, including:

Specificity/Clarity/Relevance/Coherence: These quality measures help us choose a set of topics that are neither too abstract nor overly specific. They ensure that the topics are clear and relevant to the documents that they represent. When additional information such as clustering (from LDA or other sources) and tagging (manual) documents is available, these quality criteria encourage the chosen topics to show resemblance (coherence) to those clustering/tagging in terms of transitive cover of documents they produce.

In the below, we define a variety of submodular functions that capture the above properties, and we then describe a large margin learning framework for learning convex mixtures of such components.

3.1 Submodular Components

3.1.1 Coverage Based Functions

Coverage components capture "coverage" of a set of documents.

Weighted Set Cover Function: Given a set of categories, $S \subseteq V$, define $\Gamma(S)$ as the set of documents covered — for each topic $s \in S$, $\Gamma(s) \subseteq D$ represents the documents covered by topic s and $\Gamma(S) = \cup_{s \in S} \Gamma(s)$. The weighted set cover function, defined as $f(S) = \sum_{d \in \Gamma(S)} w_d = w(\Gamma(S))$, assigns weights to the documents based on their relative importance (e.g., in Wikipedia disambiguation, the different documents could be ranked based on their priority).

Feature-based Functions: This class of function represents coverage in feature space. Given a set of categories $S \subseteq V$, and a set of features U, define $m_u(S)$ as the score associated with the set of categories S for feature $u \in U$. The feature set could represent, for example, the documents, in which case $m_u(S)$ represents the number of times document u is covered by the set S. U could also represent more complicated features. For example, in the context of Wikipedia disambiguation, U could represent TFIDF features over the documents. Feature based are then defined as $f(S) = \sum_{u \in U} \psi(m_u(S))$, where ψ is a concave (e.g., the square root) function. This function class has been successfully used in several applications (Kirchhoff and Bilmes, 2014; Wei et al., 2014a; Wei et al., 2014b).

3.1.2 Similarity based Functions

Similarity functions are defined through a similarity matrix $\mathbf{S} = \{s_{ij}\}_{i,j \in V}$. Given categories $i, j \in V$, similarity s_{ij} in our case can be defined as $s_{ij} = |\Gamma(i) \cap \Gamma(j)|$, i.e the number of documents commonly covered by both i and j.

Facility Location: The facility location function, defined as $f(S) = \sum_{i \in V} \max_{j \in S} s_{ij}$, is a natural model for k-medoids and exemplar based clustering, and has been used in several summarization problems (Tschiatschek et al., 2014; Wei et al., 2014a).

Penalty based diversity: A similarity matrix may be used to express a form of coverage of a set S but that is then penalized with a redundancy term, as in the following difference: $f(S) = \sum_{i \in V, j \in S} s_{ij} - \lambda \sum_{i \in S} \sum_{j \in S,} s_{i,j}$ (Lin and Bilmes, 2011)). Here $\lambda \in [0, 1]$. This function is submodular, but is not in general monotone, and has been used in document summarization (Lin and Bilmes, 2011), as a dispersion function (Borodin et al., 2012), and in image summarization (Tschiatschek et al., 2014).

3.1.3 Quality Control (QC) Functions

QC functions ensure a quality criteria is met by a set S of topics. We define the quality score of the set S as $F_q(S) = \sum_{s \in S} f_q(s)$, where $f_q(s)$ is the quality score of topic s for quality q. Therefore, $F_q(S)$ is a modular function in S. We investigate three types of quality control functions: Topic Specificity, Topic Clarity, and Topic Relevance.

Topic Specificity: The farther a topic is from the root of the DAG, the more specific it becomes. Topics higher up in the hierarchy are abstract and less specific. We therefore prefer topics low in the DAG, but lower topics also have less coverage. We define $f_{\text{specificity}}(s) = s_h$ where s_h is the height of topic s in the DAG. The root topic has height zero and the "height" increases as we move down the DAG in Figure 1.

Topic Clarity: Topic clarity is the fraction of descendant topics that cover one or more documents. If a topic has many descendant topics that do not cover any documents, it has less clarity. Formally, $f_{\text{clarity}}(s) = \frac{\sum_{t \in \text{descendants}(s)} [\![\Gamma(t) > 0]\!]}{|\text{descendants}(s)|}$, where $[\![.]\!]$ is the indicator function.

Topic Relevance: A topic is considered to be better related to a document if the number of hops needed to reach the document from that topic is lower. Given any set $A \subseteq D$ of document, and any topic $s \in V$, we can define $f_{\text{relevance}}(s|A) = \text{argmin}_\alpha \{\alpha : A \subseteq \Gamma^\alpha(s)\}$.

QC Functions As Barrier Modular Mixtures: We introduce a modular function for every QC function as follows

$$f^\alpha_{\text{specificity}}(s) = \begin{cases} 1 & \text{if the height of topic } s \text{ is at least } \alpha \\ 0 & \text{otherwise} \end{cases}$$

for every possible value of α. This creates a submodular mixture with as many components as the number of possible values of α. In our experiments with Wikipedia, we had α varying from 1 to 120 stepping by 1, adding 120 modular mixture components. Similarly, we define,

$$f^\beta_{\text{clarity}}(s) = \begin{cases} 1 & \text{if the clarity of topic } s \text{ is at least } \beta \\ 0 & \text{otherwise} \end{cases}$$

for every possible (discretized to make it countably finite) value of β. And,

$f^\gamma_{\text{relevance}}(s) = f_{\text{cov}}(s|\Gamma^\gamma(s))$, where $f_{\text{cov}}(.)$ is the coverage submodular function and $s|X$ indicates coverage of a topic s over a set of documents X. All these functions (modular and submodular terms) are added as mixture components in our learning framework to learn suitable weights for them. We then use these weights in our inference procedure to obtain a subset of topics as described in 3.2. We show from our experiments that this approach performs better than all other approaches and baselines.

3.1.4 Fidelity Functions

A function representing the fidelity of a set S to another reference set R is one that gets a large value when the set S represents the set R. Such a function scores inferred topics high when it resembles a reference set of topics and/or item clusters. The reference set in this case can be produced from other algorithms such as k-means, LDA and its variants or from a manually tagged corpus. Next we describe one such fidelity function.

Topic Coherence: This function scores a set of topics S high when the transitive cover (Definition 1) produced by the topics in S resembles the clusters of documents produced by an external source (k-means, LDA or manual). Given an external source that clusters the documents, producing T clusters $L_1, L_2, ..., L_T$ (for T topics), topic coherence is defined as: $f(S) = \sum_{t \in T} \max_{k \in S} w_{k,t}$ where $w_{k,t} = \text{harmonic\_mean}(w^p_{k,t}, w^r_{k,t})$ and $w^p_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|\Gamma(k)|}$ and $w^r_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|L_t|}$. Note that, $w^p_{k,t} \geq 0$ and $w^r_{k,t} \geq 0$ are the precision on recall of the resemblance and $w_{k,t}$ is the F1 measure. If the transitive cover of topics in S resembles the reference clusters L_t exactly, we attain maximum coherence (or fidelity). As the resemblance diminishes, the score decreases. The above function $f(S)$ is monotone submodular.

3.1.5 Mixture of Submodular Components:

Given the different classes of submodular functions above, we construct our submodular scoring functions $F_w(\cdot)$ as a convex combinations of these different submodular functions $f_1, f_2, \ldots, f_m$, above. In other words,

$$F_w(S) = \sum_{i=1}^{m} w_i f_i(S), \tag{2}$$

where $w = (w_1, \ldots, w_m)$, $w_i \geq 0, \sum_i w_i = 1$. The components f_i are submodular and assumed to be normalized: i.e., $f_i(\emptyset) = 0$, and $f_i(V) = 1$ for monotone functions and $\max_{A \subseteq V} f_i(A) \leq 1$ for non-monotone functions. A simple way to normalize a monotone submodular function is to define the component as $f_i(S)/f_i(V)$. This ensures that the components are *compatible* with each other. Obviously, the merit of the scoring function $F_w(\cdot)$ depends on the selection of the components.

3.2 Large Margin Learning

We optimize the weights w of the scoring function $F_w(\cdot)$ in a large-margin structured prediction framework. In this setting, we assume we have training data in the form of pairs of a set of documents, and a human generated summary as a set of topics. For example, in the case of Wikipedia disambiguation, we use the human generated disambiguation pages as the ground truth summary. We represent the set of ground-truth summaries as $\mathcal{S} = \{S_1, S_2, \cdots, S_N\}$. In large margin training, the weights are optimized such that ground-truth summaries $\mathcal{S}$ are separated from competitor summaries by a loss-dependent margin:

$$F_w(S) \geq F_w(S') + \mathcal{L}(S'), \quad \forall S \in \mathcal{S}, S' \in \mathcal{Y} \setminus \mathcal{S}, \tag{3}$$

where $\mathcal{L}(\cdot)$ is the loss function, and where $\mathcal{Y}$ is a structured output space (for example $\mathcal{Y}$ is the set of summaries that satisfy a certain budget B, i.e., $\mathcal{Y} = \{S' \subseteq V : |S'| \leq B\}$). We assume the loss to be normalized, $0 \leq \mathcal{L}(S') \leq 1, \forall S' \subseteq V$, to ensure that mixture and loss are calibrated. Equation (3) can be stated as $F_w(S) \geq \max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right], \forall S \in \mathcal{S}$ which is called *loss-augmented inference*. We introduce slack variables and minimize the regularized sum of slacks (Lin and Bilmes, 2012):

$$\min_{w \geq 0, \|w\|_1 = 1} \quad \sum_{S \in \mathcal{S}} \left[\max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right] - F_w(S)\right] + \frac{\lambda}{2}\|w\|_2^2, \tag{4}$$

where the non-negative orthant constraint, $w \geq 0$, ensures that the final mixture is submodular. Note a 2-norm regularizer is used on top of a 1-norm constraint $\|w\|_1 = 1$ which we interpret as a prior to encourage higher entropy, and thus more diverse mixture distributions. Tractability depends on the choice of the loss function. The parameters w are learnt using stochastic gradient descent as in (Tschiatschek et al., 2014).

3.3 Loss Functions

A natural choice of loss functions for our case can be derived from cluster evaluation metrics. Every inferred topic s induces a subset of documents, namely the transitive cover $\Gamma(s)$ of s. We compare these clusters with the clusters induced from the true topics in the training set and compute the loss.

In this paper, we use the Jaccard Index (JI) as a loss function. Let S be the inferred topics and T be the true topics. The Jaccard loss is defined as $\mathcal{L}_{\text{jaccard}}(S,T) = 1 - \frac{1}{k}\sum_{s \in S} \max_{t \in T} \frac{|\Gamma(s) \cap \Gamma(t)|}{|\Gamma(s) \cup \Gamma(t)|}$, where $k = |S| = |T|$ is the number of topics. When the clustering produced by the inferred and the true topics are similar, Jaccard loss is 0. When they are completely dissimilar, the loss is maximum, i.e., 1. Jaccard loss is a modular function.

3.4 Inference Algorithm: Greedy

Having learnt the weights for the mixture components, the resulting function $F_w(S) = \sum_{i=1}^{m} w_i f_i(S)$ is a submodular function. In the case when the individual components are themselves monotone (all our functions in fact are), $F_w(S)$ can be optimized by the accelerated greedy algorithm (Minoux, 1978). Thanks to submodularity, we can obtain near optimal solutions very efficiently. In case the functions are all monotone submodular, we can guarantee that the solution is within $1 - 1/e$ factor from the optimal solution.

4 Experimental Results

To validate our approach, we make use of Wikipedia category structure as a topic DAG and apply our technique to the task of automatic generation of Wikipedia disambiguation pages. We pre-processed the category graph to eliminate the cycles in order to make it a DAG. Each Wikipedia disambiguation page is manually created by Wikipedia editors by grouping a collection of Wikipedia articles into several groups. Each group is then assigned a name, which serves as a topic for the group. Typically, a disambiguation page segregates around 20-30 articles into 5-6 groups. Our goal is to measure how accurately we can recreate the groups for a disambiguation page and label them, given only the collection of articles mentioned in that disambiguation page (when actual groupings and labels are hidden).

4.1 Datasets

We parsed the contents of Wikipedia disambiguation pages and extracted disambiguation page names, article groups and group names. We collected about 8000 disambiguation pages that had at least four groups on them. Wikipedia category structure is used as the topic DAG. We eliminated few administrative categories such as "Hidden Categories", "Articles needing cleanup", and the like. The final DAG had about 1M topics and 3M links.

4.2 Evaluation Metrics

Every group of articles on the Wikipedia disambiguation page is assigned a name by the editors. Unfortunately, these names may not correspond to the Wikipedia category (topic) names. For example, one of the groups on the "Matrix" disambiguation page has a name "Business and government" and there is no Wikipedia category by that name. However, the group names generated by our (and baseline) method are from the Wikipedia categories (which forms our topic DAG). In addition, there can be multiple relevant names for a group. For example, a group on a disambiguation page may be called "Calculus", but an algorithm may rightly generate "Vector Calculus". Hence we cannot evaluate the accuracy of an algorithm just by matching the generated group names to those on the disambiguation page. To alleviate this problem, we adopt cluster-based evaluation metrics. We treat every group of articles generated by an algorithm under a topic for a disambiguation page as a cluster of articles. These are considered as *inferred* clusters for a disambiguation page. We compare them against the actual grouping of articles on the Wikipedia disambiguation page by treating those groups as *true* clusters. We can now adopt Jaccard Index, F1-measure, and NMI (Normalized Mutual Information) based cluster evaluation metrics described in (Manning et al., 2008). For each disambiguation page in the test set, we compute every metric score and then average it over all the disambiguation pages.

4.3 Methods Compared

We validated our approach by comparing against several baselines described below. We also compared two variations of our approach as described next. For each of these cases (baselines and two variations) we generated and compared the metrics (Jaccard Index, F1-measure and NMI) as described in the previous section.

KM_{docs}: K-Means algorithm run on articles as TF-IDF vectors of words. The number of clusters K is set to the number of true clusters on the Wikipedia disambiguation page.

$\text{KMed}_{\text{docs}}$: K-Medoids algorithm with articles as TF-IDF vectors of words. The number of clusters are set as in KM_{docs}.

$\text{KMed}_{\text{topics}}$: K-Medoids run on topics as TF-IDF vectors of words. The words for each topic is taken from the articles that are in the transitive cover of the topic.

LDA_{docs}: LDA algorithm with the number of topics set to the number of true clusters on the Wikipedia disambiguation page. Each article is then grouped under the highest probability topic.

SMML_{cov}: This is the submodular mixture learning case explained in section 3.1.5. Here we consider a mixture of all the submodular functions governing coverage, diversity, fidelity and QC functions. However, we exclude the similarity based functions described in section 3.1.2. Coverage based functions have a time complexity of $O(n)$ whereas similarity based functions are $O(n^2)$. By excluding similarity based functions, we can compare the quality of the results with and without $O(n^2)$ functions. We learn the mixture weights from the training set and use them during inference on the test set to subset K topics through the submodular maximization (Equation 1).

$\text{SMML}_{\text{cov+sim}}$: This case is similar to SMML_{cov} except that, we include similarity based submodular mixture components. This makes the inference time complexity $O(n^2)$.

We do not compare against HSLDA, PAM and few other techniques cited in the related work sections because they do not produce a subset of K summary topics — these are not directly comparable with our work.

4.4 Evaluation Results

We show that the submodular mixture learning and maximization approaches, i.e., SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other approaches in various metrics. In all these experiments, we performed 5 fold cross validation to learn the parameters from 80% of the disambiguation pages and evaluated on the rest of the 20%, in each fold.

In Figure 2a we summarize the results of the comparison of the methods mentioned above on Jaccard Index, F1 measure and NMI. Our proposed techniques SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other techniques consistently.

In Figures 2b and 2c we measure the number of test instances (i.e., disambiguation queries) in which each of the algorithms dominate (win) in evaluation metrics. In 60% of the disambiguation queries, SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ approaches

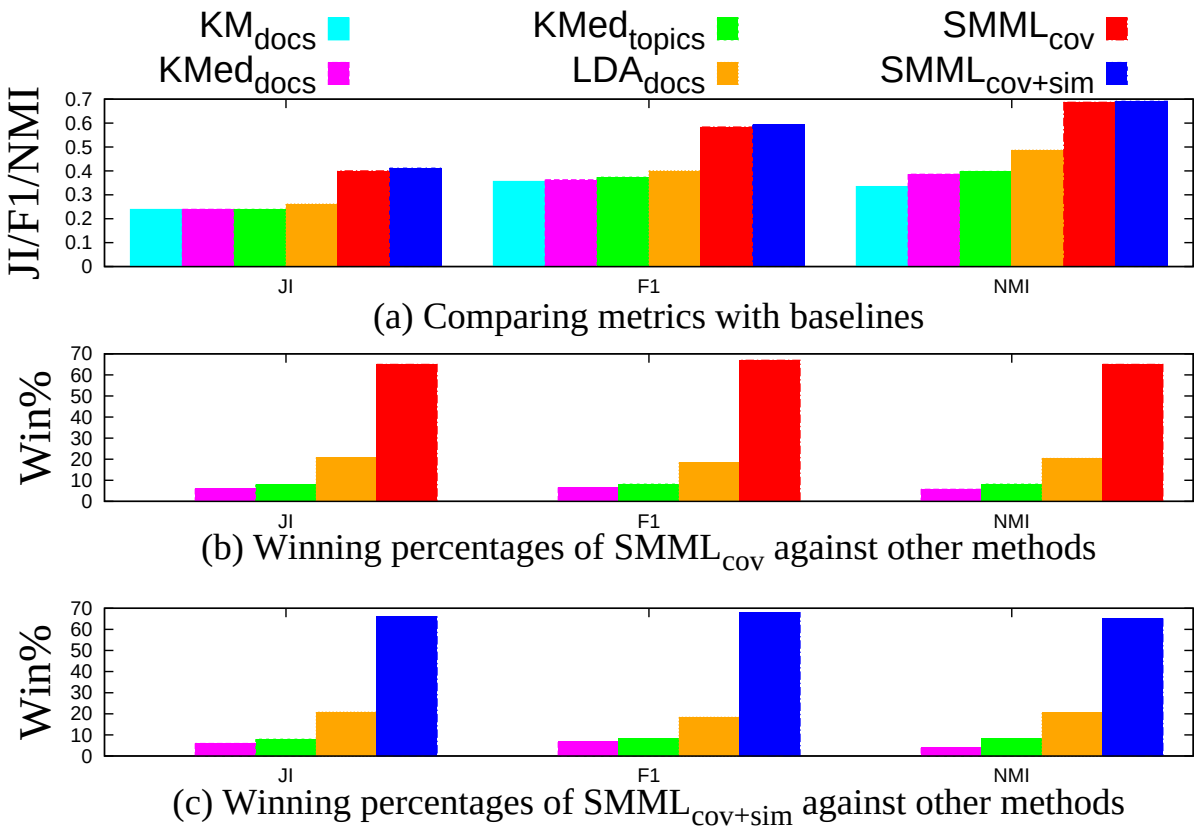

Figure 2: Comparison of techniques

produce higher JI, F1 and NMI than all other methods. This indicates that the clusters of articles produced by our technique resembles the clusters of articles present on the disambiguation page better than other techniques.

From Figures 2b and 2c it is clear that $O(n)$ time complexity based submodular mixture functions (SMML_{cov}) perform on par with $O(n^2)$ based functions ($\text{SMML}_{\text{cov+sim}}$), but at a greatly reduced execution time, demonstrating the sufficiency of $O(n)$ functions for our task. On the average, for each disambiguation query, SMML_{cov} took around 40 seconds (over 1M topics and 3M edges DAG) to infer the topics, whereas $\text{SMML}_{\text{cov+sim}}$ took around 35 minutes. Both these experiments were carried on a machine with 32 GB RAM, Eight-Core AMD Opteron(tm) Processor 2427.

5 Conclusions

We investigated a problem of summarizing topics over a massive topic DAG such that the summary set of topics produced represents the objects in the collection. This representation is characterized through various classes of submodular (and monotone) functions that captured coverage, similarity, diversity, specificity, clarity, relevance and fidelity of the topics. Currently we assume that the number of topics K is given as an input to our algorithm. It would be an interesting future problem to estimate the value of K automatically in our setting. As future work, we also plan to extend our techniques to produce a hierarchical summary of topics and scale it across heterogeneous collection of objects (from different domains) to bring all of them under the same topic DAG and investigate interesting cases thereon.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1162606, and by a Google, a Microsoft, and an Intel research award. Rishabh Iyer acknowledges support from the Microsoft Research Ph.D Fellowship.

References

Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. 2006. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, April.

W. Bi and J. T. Kwok. 2011. Multi-label classification on tree-and DAG-structured hierarchies. In *ICML*. ICML.

David M. Blei and John D. Lafferty. 2006. Correlated topic models. In *In Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. MIT Press.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, page 2003. MIT Press.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of Principles of Database Systems*, pages 155–166. ACM.

Razvan Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06), Trento, Italy*, pages 9–16, April.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. 2003. Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 178–186, New York, NY, USA. ACM.

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA.

R. Iyer, S. Jegelka, and J. Bilmes. 2013. Fast semidifferential-based submodular function optimization. ICML.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, Angela Y. Wu, Senior Member, and Senior Member. 2002. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:881–892.

D. Kempe, J. Kleinberg, and E. Tardos. 2003. Maximizing the spread of influence through a social network. In *SIGKDD*.

Katrin Kirchhoff and Jeff Bilmes. 2014. Submodularity for data selection in machine translation. October.

A. Krause and C. Guestrin. 2005. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of Uncertainity in Artificial Intelligence*. UAI.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 577–584, New York, NY, USA. ACM.

H. Lin and J. Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL*.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *The 49th Meeting of the Assoc. for Comp. Ling. Human Lang. Technologies (ACL/HLT-2011)*, Portland, OR, June.

H. Lin and J. Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, page 479490.

Hui Lin. 2012. *Submodularity in Natural Language Processing: Algorithms and Applications*. Ph.D. thesis, University of Washington, Dept. of EE.

Min ling Zhang and Zhi hua Zhou. 2007. Ml-knn: A lazy learning approach to multi-label learning. *PATTERN RECOGNITION*, 40:2007.

Arun S. Maiya, John P. Thompson, Francisco Loaiza-Lemos, and Robert M. Rolfe. 2013. Exploratory analysis of highly heterogeneous document collections. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1375–1383, New York, NY, USA. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Olena Medelyan, Ian H. Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the Wikipedia and AI workshop at AAAI-08*. AAAI.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 490–499, New York, NY, USA. ACM.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA. ACM.

David Milne. 2009. An open-source toolkit for mining wikipedia. In *In Proc. New Zealand Computer Science Research Student Conf*, page 2009.

Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, chapter 27, pages 234–243. Springer Berlin Heidelberg, Berlin/Heidelberg.

Mukund Narasimhan and Jeff Bilmes. 2004. PAC-learning bounded tree-width graphical models. In *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference (UAI-2004)*. Morgan Kaufmann Publishers, July.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294.

Adler J. Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. 2011. Hierarchically supervised latent dirichlet allocation. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *NIPS*, pages 2609–2617.

Juho Rousu, Craig Saunders, Sndor Szedmk, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626.

Jr. Silla, CarlosN. and AlexA. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Sebastian Tschiatschek, Rishabh Iyer, Hoachen Wei, and Jeff Bilmes. 2014. Learning Mixtures of Submodular Functions for Image Collection Summarization. In *Neural Information Processing Systems (NIPS)*.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer US.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2014a. Fast multi-stage submodular maximization. In *ICML*.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, Chris Bartels, and Jeff Bilmes. 2014b. Submodular subset selection for large-scale speech training data. *Proceedings of ICASSP, Florence, Italy*.

Torsten Zesch and Iryna Gurevych. 2007. Analysis of the wikipedia category graph for nlp applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, pages 1–8, Rochester, April. Association for Computational Linguistics.

Summarization of Multi-Document Topic Hierarchies using Submodular Mixtures

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, 40076, India
bairi@cse.iitb.ac.in

Rishabh Iyer
University of Washington
Seattle, WA-98175, USA
rkiyer@u.washington.edu

Ganesh Ramakrishnan
IIT Bombay
Mumbai, 40076, India
ganesh@cse.iitb.ac.in

Jeff Bilmes
University of Washington
Seattle, WA-98175, USA
bilmes@uw.edu

Abstract

We study the problem of summarizing DAG-structured topic hierarchies over a given set of documents. Example applications include automatically generating Wikipedia disambiguation pages for a set of articles, and generating candidate multi-labels for preparing machine learning datasets (e.g., for text classification, functional genomics, and image classification). Unlike previous work, which focuses on clustering the set of documents using the topic hierarchy as features, we directly pose the problem as a submodular optimization problem on a topic hierarchy using the documents as features. Desirable properties of the chosen topics include document coverage, specificity, topic diversity, and topic homogeneity, each of which, we show, is naturally modeled by a submodular function. Other information, provided say by unsupervised approaches such as LDA and its variants, can also be utilized by defining a submodular function that expresses coherence between the chosen topics and this information. We use a large-margin framework to learn convex mixtures over the set of submodular components. We empirically evaluate our method on the problem of automatically generating Wikipedia disambiguation pages using human generated clusterings as ground truth. We find that our framework improves upon several baselines according to a variety of standard evaluation metrics including the Jaccard Index, F1 score and NMI, and moreover, can be scaled to extremely large scale problems.

1 Introduction

Several real world machine learning applications involve hierarchy based categorization of topics for a set of objects. Objects could be, e.g., a set of documents for text classification, a set of genes in functional genomics, or a set of images in computer vision. One can often define a natural topic hierarchy to categorize these objects. For example, in text and image classification problems, each document or image is assigned a hierarchy of labels — a baseball page is assigned the labels "baseball" and "sports." Moreover, many of these applications, naturally have an existing topic hierarchy generated on the entire set of objects (Rousu et al., 2006; Barutcuoglu et al., 2006; ling Zhang and hua Zhou, 2007; Silla and Freitas, 2011; Tsoumakas et al., 2010).

Given a DAG-structured topic hierarchy and a subset of objects, we investigate the problem of finding a subset of DAG-structured topics that are induced by that subset (of objects). This problem arises naturally in several real world applications. For example, consider the problem of identifying appropriate label sets for a collection of articles. Several existing text collection datasets such as 20 Newsgroup[1], Reuters-21578[2] work with a predefined set of topics. We observe that these topic names are highly abstract[3] for the articles categorized under them. On the other hand, techniques proposed by systems such as Wikipedia Miner (Milne, 2009) and TAGME (Ferragina and Scaiella, 2010) generate several labels for each article in the dataset that are highly specific to the article. Collating all labels from all articles to create a label

[1] http://qwone.com/~jason/20Newsgroups/

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[3] Topic *Concept* is more abstract than the topic *Science* which is more abstract than the topic*Chemistry*

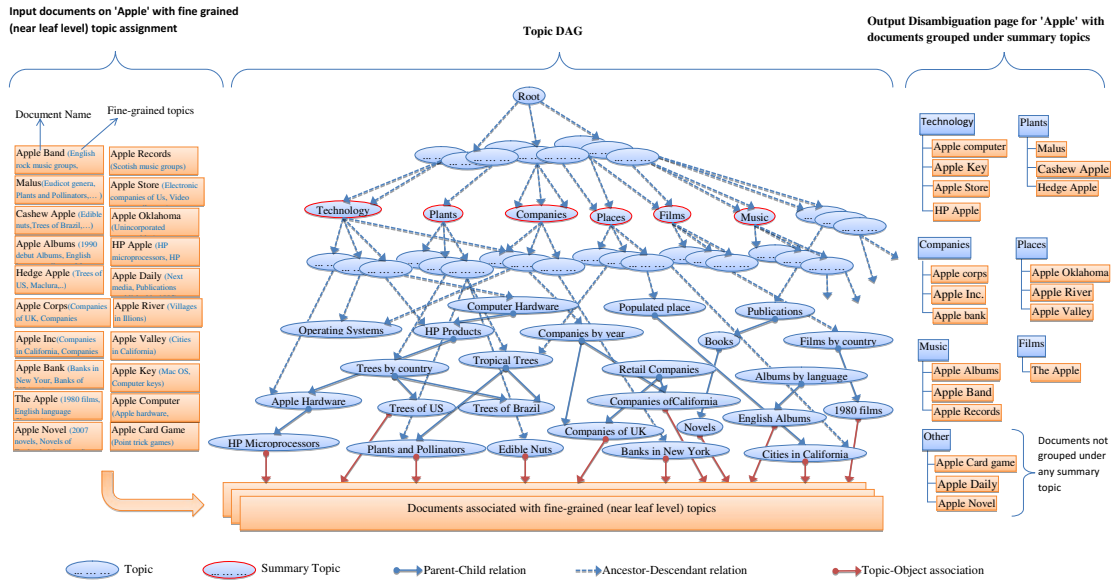

Figure 1: Topic Summarization overview. On the left, we show many documents related to Apple. In the middle, a Wikipedia category hierarchy shown as a topic DAG, links these documents at the leaf level. On the right, we show the output of our summarization process, which creates a set of summary topics (Plants, Technology, Companies, Films, Music and Places in this example) with the input documents classified under them.

set for the dataset can result in a large number of labels and become unmanageable. Our proposed techniques can summarize such large sets of labels into a smaller and more meaningful label sets using a DAG-structured topic hierarchy. This also holds for image classification problems and datasets like ImageNet (Deng et al., 2009). We use the term *summarize* to highlight the fact that the smaller label set semantically covers the larger label set. For example, the topics *Physics*, *Chemistry*, and *Mathematics* can be summarized into a topic *Science*.

A particularly important application of our work (and the one we use for our evaluations in Section 4) is the following: Given a collection of articles spanning different topics, but with similar titles, automatically generate a disambiguation page for those titles using the Wikipedia category hierarchy[4] as a topic DAG. Disambiguation pages[5] on Wikipedia are used to resolve conflicts in article titles that occur when a title is naturally associated with multiple articles on distinct topics. Each disambiguation page organizes articles into several groups, where the articles in each group pertain only to a specific topic. Disambiguations may be seen as paths in a hierarchy leading to different articles that arguably could have the same title. For example, the title *Apple*[6] can refer to a plant, a company, a film, a television show, a place, a technology, an album, a record label, and a newspaper daily. The problem then, is to organize the articles into multiple groups where each group contains articles of similar nature (topics) and has an appropriately discerned group heading. Figure 1 describes the topic summarization process for creation of the disambiguation page for "Apple".

All the above mentioned problems can be modeled as the problem of finding the most representative subset of topic nodes from a DAG-Structured topic hierarchy. We argue that many formulations of this problem are natural instances of submodular maximization, and provide a learning framework to create submodular mixtures to solve this problem. A set function $\mathrm{f}(.)$ is said to be submodular if for any element v and sets $A \subseteq B \subseteq V \setminus \{v\}$, where V represents the ground set of elements, $\mathrm{f}(A \cup \{v\}) - \mathrm{f}(A) \geq \mathrm{f}(B \cup \{v\}) - \mathrm{f}(B)$. This is called the diminishing returns property and states, informally, that adding an element to a smaller set increases the function value more than adding that element to a larger set. Submodular functions naturally model notions of coverage and diversity in applications, and therefore, a number of machine learning problems can be modeled as forms of submodular optimization (Kempe et al., 2003; Krause and Guestrin, 2005; Narasimhan and Bilmes, 2004; Iyer et al., 2013; Lin and Bilmes, 2012; Lin and Bilmes, 2010). In this paper, we investigate structured prediction methods for learn-

[4] http://en.wikipedia.org/wiki/Help:Categories

[5] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation

[6] http://en.wikipedia.org/wiki/Apple_(disambiguation)

ing weighted mixtures of submodular functions to summarize topics for a collection of objects using DAG-structured topic hierarchies. Throughout this paper we use the terms "topic" and "category" interchangeably.

1.1 Related Work

To the best of our knowledge, the specific problem we consider here is new. Previous work on identifying topics can be broadly categorized into one of the following types: a) cluster the objects and then identify names for the clusters; or b) dynamically identify topics (including hierarchical) for a set of objects. LDA (Blei et al., 2003) clusters the documents and simultaneously produces a set of topics into which the documents are clustered. In LDA, each document may be viewed as a mixture of various topics and the topic distribution is assumed to have a Dirichlet prior. LDA associates a group of high probability words to each identified topic. A name can be assigned to a topic by manually inspecting the words or using additional algorithms like (Mei et al., 2007; Maiya et al., 2013). LDA does not make use of existing topic hierarchies and correlation between topics. The Correlated Topic Model (Blei and Lafferty, 2006) induces a correlation structure between topics by using the logistic normal distribution instead of the Dirichlet. Another extension is the hierarchical LDA (Blei et al., 2004), where topics are joined together in a hierarchy by using the nested Chinese restaurant process. Nonparametric extensions of LDA include the Hierarchical Dirichlet Process (Teh et al., 2006) mixture model, which allows the number of topics to be unbounded and learnt from data and the Nested Chinese Restaurant Process which allows topics to be arranged in a hierarchy whose structure is learnt from data. In each of these approaches, unlike our proposed approach, an existing topic hierarchy is not used, nor is any additional object-topic information leveraged.

The pachinko allocation model (PAM)(Li and McCallum, 2006) captures arbitrary, nested, and possibly sparse correlations between topics using a DAG. The leaves of the DAG represent individual words in the vocabulary, while each interior node represents a correlation among its children, which may be words or other interior nodes (topics). PAM learns the probability distributions of words in a topic, subtopics in a topic, and topics in a document. We cannot, however, generate a subset of topics from a large existing topic DAG that can act as summary topics, using PAM.

HSLDA (Perotte et al., 2011) introduces a hierarchically supervised LDA model to infer hierarchical labels for a document. It assumes an existing label hierarchy in the form of a tree. The model infers one or more labels such that, if a label l is inferred as relevant to a document, then all the labels from l to the root of the tree are also inferred as relevant to the document. Our approach differs from HSLDA since: (1) we use the label hierarchy to infer a set of labels for a *group* of documents; (2) we do not enforce the label hierarchy to be a tree as it can be a DAG; and (3) generalizing HSLDA to use a DAG structured hierarchy and infer labels for a *group* of documents (e.g., combining into one big document) also may not help in solving our problem. HSLDA will apply *all* the relevant labels to the documents as per the classifier that it learns for every label. Moreover, the "root" label is always applied and it is very likely that many labels near the top level of the label hierarchy are also classified as relevant to the group of documents.

Wei and James (Bi and Kwok, 2011) present a hierarchical multi-label classification algorithm that can be used on both tree and DAG structured hierarchies. They formulate a search for the optimal consistent multi-label as the finding of the best subgraph in a tree/DAG. In our approach, we assume, individual documents are already associated with one or more topics and we find a consistent label set for a group of documents using the DAG structured topic hierarchy.

Medelyan et al. (Medelyan et al., 2008) and Ferragina et al. (Ferragina and Scaiella, 2010) detect topics for a document using Wikipedia article names and category names as the topic vocabulary. These systems are able to extract signals from a text document and identify Wikipedia articles and/or categories that optimally match the document and assign those article/category names as topics for the document. When run on a large collection of documents, these approaches generate enormous numbers of topics, a problem our proposed approach addresses.

1.2 Our Contributions

While most prior work discussed above focuses on the underlying set of documents, (*e.g.*, by clustering documents), we focus directly on the topics. In particular, we formulate the problem as subset selection on the set of topics within a DAG while simultaneously considering the documents to be categorized. Our method can scale to the colossal size of the DAG (1 million topics and 3 million correlation links between topics in Wikipedia). Moreover, our approach can

naturally incorporate outputs from many of the aforementioned algorithms. Our approach is based on submodular maximization and mixture learning, which has been successfully used in applications such as document summarization (Lin, 2012) and image summarization (Tschiatschek et al., 2014), but has never been applied to topic identification tasks or, more generally, DAG summarization.

We introduce a family of submodular functions to identify an appropriate set of topics from a DAG structured hierarchy of topics for a group of documents. We characterize this topic appropriateness through a set of desirable properties such as coverage, diversity, specificity, clarity, and relevance. Each of the submodular function components we consider are monotone, thereby ensuring a near optimal performance obtainable via a simple greedy algorithm for optimization.[7]. We also show how our technique naturally embodies outputs of other algorithms such as LDA, clustering, and classifications. Finally, we utilize a large margin formulation for learning mixtures of these submodular functions, and show how we can optimally learn them from training data.

Our approach demonstrates how to utilize the features collectively in the document space and the topic space to infer a set of topics. From an empirical perspective, we introduce and evaluate our approach on a dataset of around 8000 disambiguations that was extracted from Wikipedia and subsequently cleaned using the methods described in the experimentation section. We show that our learning framework outperforms many of the baselines, and is practical enough to be used on large corpora.

2 Problem Formulation

Let $G\,(V, E)$ be the DAG structured topic hierarchy with V topics. These topics are observed to have a parent child (*isa*) relationship forming a DAG. Let D be the set of documents that are associated with one or more of these topics. The middle portion of Figure 1 depicts a topic hierarchy with associated documents. The association links between the documents and topics can be hard or soft. In case of a hard link, a document is attached to a set of topics. Examples include multi-labeled documents. In case of a soft link, a document is associated with a topic with some degree of confidence (or probability). Furthermore, if a document is attached to a topic t, we assume that all the ancestor topics of t are also relevant for that document. This assumption has been employed in earlier works (Blei et al., 2004; Bi and Kwok, 2011; Rousu et al., 2006) as well. Given a budget of K, our objective is to choose a set of K topics from V, which best describe the documents in D. The notion of best describing topics is characterized through a set of desirable properties - coverage, diversity, specificity, clarity, relevance and fidelity - that K topics have to satisfy. The submodular functions that we introduce in the next section ensure these properties are satisfied. Formally, we solve the following discrete optimization problem:

$$S^* \in \underset{S \subseteq V : |S| \leq K}{\operatorname{argmax}} \sum_i w_i f_i(S) \qquad (1)$$

where, f_i are monotone submodular mixture components and $w_i \geq 0$ are the weights associated with those mixture components. Set S^* is the summary topics scored best.

It is easy to find massive (i.e., size in the order of million) DAG structured topic hierarchies in practice. Wikipedia's category hierarchy consists of more than 1M categories (topics) arranged hierarchically. In fact, they form a cyclic graph (Zesch and Gurevych, 2007). However, we can convert it to a DAG by eliminating the cycles as described in the supplementary material. YAGO (Suchanek et al., 2007) and Freebase (Bollacker et al., 2008) are other instances of massive topic hierarchies. The association of the documents with the existing topic hierarchy is also well studied. Systems such as WikipediaMiner (Milne, 2009), TAGME (Ferragina and Scaiella, 2010) and several annotation systems such as (Dill et al., 2003; Mihalcea and Csomai, 2007; Bunescu and Pasca, 2006) attach topics from Wikipedia (and other catalogs) to the documents by establishing the hard or soft links mentioned above.

Our goal is the following: Given a (ground set) collection V of topics organized in a pre-existing hierarchical DAG structure, and a collection D of documents, chose a size $K \in \mathbb{Z}_+$ representative subset of topics. Our approach is distinct from earlier work (e.g., (Kanungo et al., 2002; Blei et al., 2003)) where typically only a set of documents is classified and categorized in some way. We next provide a few definitions needed later in the paper.

Definition 1: Transitive Cover Γ): A topic t is said to cover a set of documents $\Gamma(t)$, called the transitive cover of the topic t, if for all documents $i \in \Gamma(t)$, either i is associated directly with topic t or with any of the descendant topics of t in the topic DAG. A natural extension of this definition to a set of topics T is defined as $\Gamma(T) = \cup_{t \in T} \Gamma(t)$.

[7] A simple greedy algorithm (Nemhauser et al., 1978) obtains a $1 - 1/e$ approximation guarantee for monotone submodular function maximization

Definition 2: Truncated Transitive Cover (Γ^α): This is a transitive cover of topic t, but with the limitation that the path length between a document and the topic t is not more than α. Hence, $|\Gamma^\alpha(t)| \leq |\Gamma(t)|$.

While our problem is closely related to clustering approaches, which consider the set of documents directly, there are some crucial differences. In particular, we focus on producing a clustering of documents where clusters are encouraged to honor a pre-defined DAG structured topic hierarchy. Existing agglomerative clustering algorithms focusing on the coverage of documents may not produce the desired clustering. To understand this, consider six documents d1, d2 ... d6 to be grouped into three clusters. There may be multiple ways to do this depending upon multiple aggregation paths present in the topic DAG: ((d1, d2), (d3, d4), (d5, d6)) or ((d1, d2, d3), (d4, d5), (d6)) or ((d1, d2, d3, d4), (d5), (d6)) or something else. Hence, we need more stringent measures to prefer one clustering over the others. Our work addresses this with a variety of quality criteria (coverage, diversity, specificity, clarity, relevance and fidelity, which are explained later in this paper) that are organically derived from well established submodular functions. And, most importantly, we learn the right mixture of these qualities to be enforced from the data itself. Furthermore, our approach also generalizes these clustering approaches, since one of the components in our mixture of submodular functions is defined via these unsupervised approaches, and maps a given clustering to a set of topics in the hierarchy.

3 Submodular Components and Learning

Summarization is the task of extracting information from a source that is both small in size but still representative. Our problem is different from traditional summarization tasks since we have an underlying DAG as a topic hierarchy that we wish to summarize in response to a subset of documents. Thus, a critical part of our problem is to take the graph structure into account while creating the summaries. Below, we identify properties we wish our summaries to posses.

Coverage: A summary set of topics should cover most of the documents. A document is said to be covered by a topic if there exists a path from the topic, going through intermediary descendant topics, to the document, i.e., the document is within the transitive cover of the topic.

Diversity: Summaries should be as diverse as possible, i.e., each summary topic should cover a unique set of documents. When a document is covered by more than one topic, that document is redundantly covered, e.g., "Finance" and "Banking" would be unlikely members of the same summary.

Summary qualities also involve "quality" notions, including:

Specificity/Clarity/Relevance/Coherence: These quality measures help us choose a set of topics that are neither too abstract nor overly specific. They ensure that the topics are clear and relevant to the documents that they represent. When additional information such as clustering (from LDA or other sources) and tagging (manual) documents is available, these quality criteria encourage the chosen topics to show resemblance (coherence) to those clustering/tagging in terms of transitive cover of documents they produce.

In the below, we define a variety of submodular functions that capture the above properties, and we then describe a large margin learning framework for learning convex mixtures of such components.

3.1 Submodular Components

3.1.1 Coverage Based Functions

Coverage components capture "coverage" of a set of documents.

Weighted Set Cover Function: Given a set of categories, $S \subseteq V$, define $\Gamma(S)$ as the set of documents covered — for each topic $s \in S$, $\Gamma(s) \subseteq D$ represents the documents covered by topic s and $\Gamma(S) = \cup_{s \in S} \Gamma(s)$. The weighted set cover function, defined as $f(S) = \sum_{d \in \Gamma(S)} w_d = w(\Gamma(S))$, assigns weights to the documents based on their relative importance (e.g., in Wikipedia disambiguation, the different documents could be ranked based on their priority).

Feature-based Functions: This class of function represents coverage in feature space. Given a set of categories $S \subseteq V$, and a set of features U, define $m_u(S)$ as the score associated with the set of categories S for feature $u \in U$. The feature set could represent, for example, the documents, in which case $m_u(S)$ represents the number of times document u is covered by the set S. U could also represent more complicated features. For example, in the context of Wikipedia disambiguation, U could represent TFIDF features over the documents. Feature based are then defined as $f(S) = \sum_{u \in U} \psi(m_u(S))$, where ψ is a concave (e.g., the square root) function. This function class has been successfully used in several applications (Kirchhoff and Bilmes, 2014; Wei et al., 2014a; Wei et al., 2014b).

3.1.2 Similarity based Functions

Similarity functions are defined through a similarity matrix $\mathbf{S} = \{s_{ij}\}_{i,j \in V}$. Given categories $i, j \in V$, similarity s_{ij} in our case can be defined as $s_{ij} = |\Gamma(i) \cap \Gamma(j)|$, i.e the number of documents commonly covered by both i and j.

Facility Location: The facility location function, defined as $f(S) = \sum_{i \in V} \max_{j \in S} s_{ij}$, is a natural model for k-medoids and exemplar based clustering, and has been used in several summarization problems (Tschiatschek et al., 2014; Wei et al., 2014a).

Penalty based diversity: A similarity matrix may be used to express a form of coverage of a set S but that is then penalized with a redundancy term, as in the following difference: $f(S) = \sum_{i \in V, j \in S} s_{ij} - \lambda \sum_{i \in S} \sum_{j \in S,} s_{i,j}$ (Lin and Bilmes, 2011)). Here $\lambda \in [0, 1]$. This function is submodular, but is not in general monotone, and has been used in document summarization (Lin and Bilmes, 2011), as a dispersion function (Borodin et al., 2012), and in image summarization (Tschiatschek et al., 2014).

3.1.3 Quality Control (QC) Functions

QC functions ensure a quality criteria is met by a set S of topics. We define the quality score of the set S as $F_q(S) = \sum_{s \in S} f_q(s)$, where $f_q(s)$ is the quality score of topic s for quality q. Therefore, $F_q(S)$ is a modular function in S. We investigate three types of quality control functions: Topic Specificity, Topic Clarity, and Topic Relevance.

Topic Specificity: The farther a topic is from the root of the DAG, the more specific it becomes. Topics higher up in the hierarchy are abstract and less specific. We therefore prefer topics low in the DAG, but lower topics also have less coverage. We define $f_{\text{specificity}}(s) = s_h$ where s_h is the height of topic s in the DAG. The root topic has height zero and the "height" increases as we move down the DAG in Figure 1.

Topic Clarity: Topic clarity is the fraction of descendant topics that cover one or more documents. If a topic has many descendant topics that do not cover any documents, it has less clarity. Formally, $f_{\text{clarity}}(s) = \frac{\sum_{t \in \text{descendants}(s)} [\![\Gamma(t) > 0]\!]}{|\text{descendants}(s)|}$, where $[\![.]\!]$ is the indicator function.

Topic Relevance: A topic is considered to be better related to a document if the number of hops needed to reach the document from that topic is lower. Given any set $A \subseteq D$ of document, and any topic $s \in V$, we can define $f_{\text{relevance}}(s|A) = \text{argmin}_\alpha \{\alpha : A \subseteq \Gamma^\alpha(s)\}$.

QC Functions As Barrier Modular Mixtures: We introduce a modular function for every QC function as follows

$$f^\alpha_{\text{specificity}}(s) = \begin{cases} 1 & \text{if the height of topic } s \text{ is at least } \alpha \\ 0 & \text{otherwise} \end{cases}$$

for every possible value of α. This creates a submodular mixture with as many components as the number of possible values of α. In our experiments with Wikipedia, we had α varying from 1 to 120 stepping by 1, adding 120 modular mixture components. Similarly, we define,

$$f^\beta_{\text{clarity}}(s) = \begin{cases} 1 & \text{if the clarity of topic } s \text{ is at least } \beta \\ 0 & \text{otherwise} \end{cases}$$

for every possible (discretized to make it countably finite) value of β. And,

$f^\gamma_{\text{relevance}}(s) = f_{\text{cov}}(s|\Gamma^\gamma(s))$, where $f_{\text{cov}}(.)$ is the coverage submodular function and $s|X$ indicates coverage of a topic s over a set of documents X. All these functions (modular and submodular terms) are added as mixture components in our learning framework to learn suitable weights for them. We then use these weights in our inference procedure to obtain a subset of topics as described in 3.2. We show from our experiments that this approach performs better than all other approaches and baselines.

3.1.4 Fidelity Functions

A function representing the fidelity of a set S to another reference set R is one that gets a large value when the set S represents the set R. Such a function scores inferred topics high when it resembles a reference set of topics and/or item clusters. The reference set in this case can be produced from other algorithms such as k-means, LDA and its variants or from a manually tagged corpus. Next we describe one such fidelity function.

Topic Coherence: This function scores a set of topics S high when the transitive cover (Definition 1) produced by the topics in S resembles the clusters of documents produced by an external source (k-means, LDA or manual). Given an external source that clusters the documents, producing T clusters $L_1, L_2, ..., L_T$ (for T topics), topic coherence is defined as: $f(S) = \sum_{t \in T} \max_{k \in S} w_{k,t}$ where $w_{k,t} = \text{harmonic\_mean}(w^p_{k,t}, w^r_{k,t})$ and $w^p_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|\Gamma(k)|}$ and $w^r_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|L_t|}$. Note that, $w^p_{k,t} \geq 0$ and $w^r_{k,t} \geq 0$ are the precision on recall of the resemblance and $w_{k,t}$ is the F1 measure. If the transitive cover of topics in S resembles the reference clusters L_t exactly, we attain maximum coherence (or fidelity). As the resemblance diminishes, the score decreases. The above function $f(S)$ is monotone submodular.

3.1.5 Mixture of Submodular Components:

Given the different classes of submodular functions above, we construct our submodular scoring functions $F_w(\cdot)$ as a convex combinations of these different submodular functions $f_1, f_2, \ldots, f_m$, above. In other words,

$$F_w(S) = \sum_{i=1}^{m} w_i f_i(S), \tag{2}$$

where $w = (w_1, \ldots, w_m)$, $w_i \geq 0, \sum_i w_i = 1$. The components f_i are submodular and assumed to be normalized: i.e., $f_i(\emptyset) = 0$, and $f_i(V) = 1$ for monotone functions and $\max_{A \subseteq V} f_i(A) \leq 1$ for non-monotone functions. A simple way to normalize a monotone submodular function is to define the component as $f_i(S)/f_i(V)$. This ensures that the components are *compatible* with each other. Obviously, the merit of the scoring function $F_w(\cdot)$ depends on the selection of the components.

3.2 Large Margin Learning

We optimize the weights w of the scoring function $F_w(\cdot)$ in a large-margin structured prediction framework. In this setting, we assume we have training data in the form of pairs of a set of documents, and a human generated summary as a set of topics. For example, in the case of Wikipedia disambiguation, we use the human generated disambiguation pages as the ground truth summary. We represent the set of ground-truth summaries as $\mathcal{S} = \{S_1, S_2, \cdots, S_N\}$. In large margin training, the weights are optimized such that ground-truth summaries $\mathcal{S}$ are separated from competitor summaries by a loss-dependent margin:

$$F_w(S) \geq F_w(S') + \mathcal{L}(S'), \quad \forall S \in \mathcal{S}, S' \in \mathcal{Y} \setminus \mathcal{S}, \tag{3}$$

where $\mathcal{L}(\cdot)$ is the loss function, and where $\mathcal{Y}$ is a structured output space (for example $\mathcal{Y}$ is the set of summaries that satisfy a certain budget B, i.e., $\mathcal{Y} = \{S' \subseteq V : |S'| \leq B\}$). We assume the loss to be normalized, $0 \leq \mathcal{L}(S') \leq 1, \forall S' \subseteq V$, to ensure that mixture and loss are calibrated. Equation (3) can be stated as $F_w(S) \geq \max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right], \forall S \in \mathcal{S}$ which is called *loss-augmented inference*. We introduce slack variables and minimize the regularized sum of slacks (Lin and Bilmes, 2012):

$$\min_{w \geq 0, \|w\|_1 = 1} \quad \sum_{S \in \mathcal{S}} \left[\max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right] - F_w(S)\right] + \frac{\lambda}{2}\|w\|_2^2, \tag{4}$$

where the non-negative orthant constraint, $w \geq 0$, ensures that the final mixture is submodular. Note a 2-norm regularizer is used on top of a 1-norm constraint $\|w\|_1 = 1$ which we interpret as a prior to encourage higher entropy, and thus more diverse mixture distributions. Tractability depends on the choice of the loss function. The parameters w are learnt using stochastic gradient descent as in (Tschiatschek et al., 2014).

3.3 Loss Functions

A natural choice of loss functions for our case can be derived from cluster evaluation metrics. Every inferred topic s induces a subset of documents, namely the transitive cover $\Gamma(s)$ of s. We compare these clusters with the clusters induced from the true topics in the training set and compute the loss.

In this paper, we use the Jaccard Index (JI) as a loss function. Let S be the inferred topics and T be the true topics. The Jaccard loss is defined as $\mathcal{L}_{\text{jaccard}}(S,T) = 1 - \frac{1}{k}\sum_{s \in S} \max_{t \in T} \frac{|\Gamma(s) \cap \Gamma(t)|}{|\Gamma(s) \cup \Gamma(t)|}$, where $k = |S| = |T|$ is the number of topics. When the clustering produced by the inferred and the true topics are similar, Jaccard loss is 0. When they are completely dissimilar, the loss is maximum, i.e., 1. Jaccard loss is a modular function.

3.4 Inference Algorithm: Greedy

Having learnt the weights for the mixture components, the resulting function $F_w(S) = \sum_{i=1}^{m} w_i f_i(S)$ is a submodular function. In the case when the individual components are themselves monotone (all our functions in fact are), $F_w(S)$ can be optimized by the accelerated greedy algorithm (Minoux, 1978). Thanks to submodularity, we can obtain near optimal solutions very efficiently. In case the functions are all monotone submodular, we can guarantee that the solution is within $1 - 1/e$ factor from the optimal solution.

4 Experimental Results

To validate our approach, we make use of Wikipedia category structure as a topic DAG and apply our technique to the task of automatic generation of Wikipedia disambiguation pages. We pre-processed the category graph to eliminate the cycles in order to make it a DAG. Each Wikipedia disambiguation page is manually created by Wikipedia editors by grouping a collection of Wikipedia articles into several groups. Each group is then assigned a name, which serves as a topic for the group. Typically, a disambiguation page segregates around 20-30 articles into 5-6 groups. Our goal is to measure how accurately we can recreate the groups for a disambiguation page and label them, given only the collection of articles mentioned in that disambiguation page (when actual groupings and labels are hidden).

4.1 Datasets

We parsed the contents of Wikipedia disambiguation pages and extracted disambiguation page names, article groups and group names. We collected about 8000 disambiguation pages that had at least four groups on them. Wikipedia category structure is used as the topic DAG. We eliminated few administrative categories such as "Hidden Categories", "Articles needing cleanup", and the like. The final DAG had about 1M topics and 3M links.

4.2 Evaluation Metrics

Every group of articles on the Wikipedia disambiguation page is assigned a name by the editors. Unfortunately, these names may not correspond to the Wikipedia category (topic) names. For example, one of the groups on the "Matrix" disambiguation page has a name "Business and government" and there is no Wikipedia category by that name. However, the group names generated by our (and baseline) method are from the Wikipedia categories (which forms our topic DAG). In addition, there can be multiple relevant names for a group. For example, a group on a disambiguation page may be called "Calculus", but an algorithm may rightly generate "Vector Calculus". Hence we cannot evaluate the accuracy of an algorithm just by matching the generated group names to those on the disambiguation page. To alleviate this problem, we adopt cluster-based evaluation metrics. We treat every group of articles generated by an algorithm under a topic for a disambiguation page as a cluster of articles. These are considered as *inferred* clusters for a disambiguation page. We compare them against the actual grouping of articles on the Wikipedia disambiguation page by treating those groups as *true* clusters. We can now adopt Jaccard Index, F1-measure, and NMI (Normalized Mutual Information) based cluster evaluation metrics described in (Manning et al., 2008). For each disambiguation page in the test set, we compute every metric score and then average it over all the disambiguation pages.

4.3 Methods Compared

We validated our approach by comparing against several baselines described below. We also compared two variations of our approach as described next. For each of these cases (baselines and two variations) we generated and compared the metrics (Jaccard Index, F1-measure and NMI) as described in the previous section.

KM_{docs}: K-Means algorithm run on articles as TF-IDF vectors of words. The number of clusters K is set to the number of true clusters on the Wikipedia disambiguation page.

$\text{KMed}_{\text{docs}}$: K-Medoids algorithm with articles as TF-IDF vectors of words. The number of clusters are set as in KM_{docs}.

$\text{KMed}_{\text{topics}}$: K-Medoids run on topics as TF-IDF vectors of words. The words for each topic is taken from the articles that are in the transitive cover of the topic.

LDA_{docs}: LDA algorithm with the number of topics set to the number of true clusters on the Wikipedia disambiguation page. Each article is then grouped under the highest probability topic.

SMML_{cov}: This is the submodular mixture learning case explained in section 3.1.5. Here we consider a mixture of all the submodular functions governing coverage, diversity, fidelity and QC functions. However, we exclude the similarity based functions described in section 3.1.2. Coverage based functions have a time complexity of $O(n)$ whereas similarity based functions are $O(n^2)$. By excluding similarity based functions, we can compare the quality of the results with and without $O(n^2)$ functions. We learn the mixture weights from the training set and use them during inference on the test set to subset K topics through the submodular maximization (Equation 1).

$\text{SMML}_{\text{cov+sim}}$: This case is similar to SMML_{cov} except that, we include similarity based submodular mixture components. This makes the inference time complexity $O(n^2)$.

We do not compare against HSLDA, PAM and few other techniques cited in the related work sections because they do not produce a subset of K summary topics — these are not directly comparable with our work.

4.4 Evaluation Results

We show that the submodular mixture learning and maximization approaches, i.e., SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other approaches in various metrics. In all these experiments, we performed 5 fold cross validation to learn the parameters from 80% of the disambiguation pages and evaluated on the rest of the 20%, in each fold.

In Figure 2a we summarize the results of the comparison of the methods mentioned above on Jaccard Index, F1 measure and NMI. Our proposed techniques SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other techniques consistently.

In Figures 2b and 2c we measure the number of test instances (i.e., disambiguation queries) in which each of the algorithms dominate (win) in evaluation metrics. In 60% of the disambiguation queries, SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ approaches

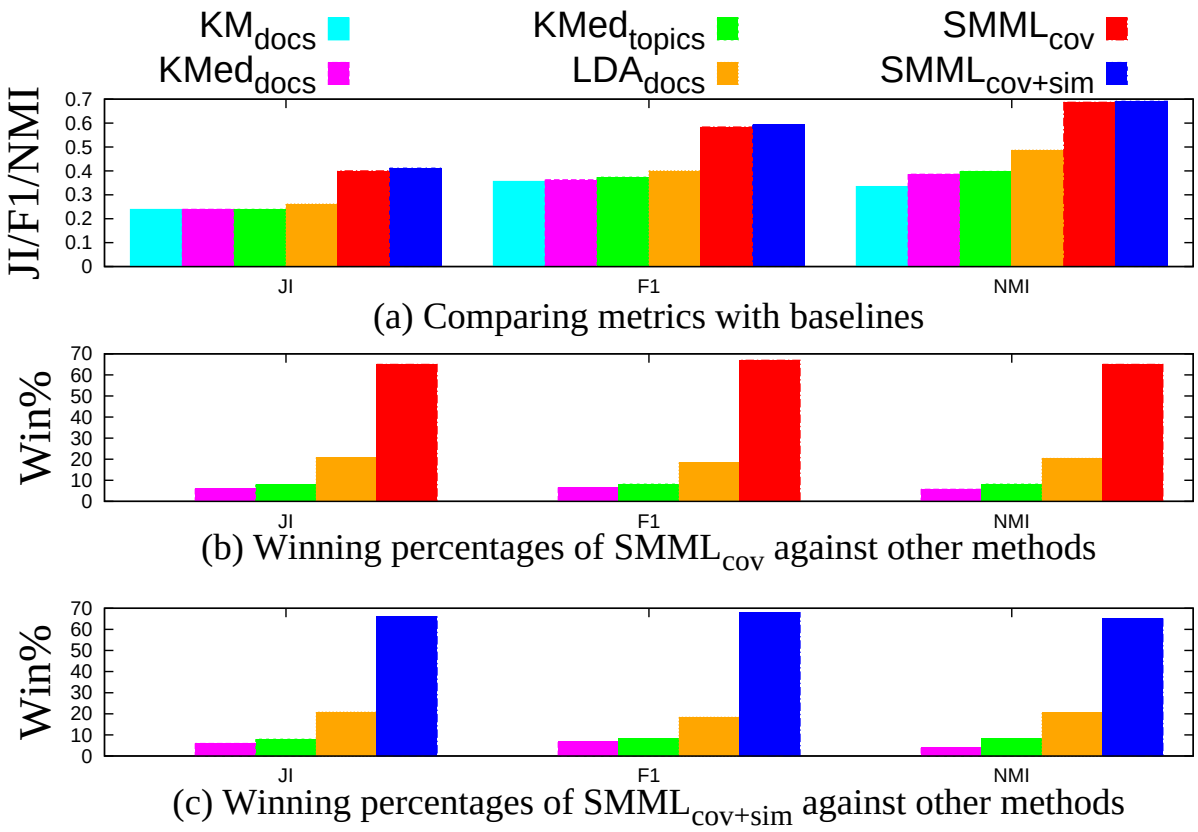

Figure 2: Comparison of techniques

produce higher JI, F1 and NMI than all other methods. This indicates that the clusters of articles produced by our technique resembles the clusters of articles present on the disambiguation page better than other techniques.

From Figures 2b and 2c it is clear that $O(n)$ time complexity based submodular mixture functions (SMML_{cov}) perform on par with $O(n^2)$ based functions ($\text{SMML}_{\text{cov+sim}}$), but at a greatly reduced execution time, demonstrating the sufficiency of $O(n)$ functions for our task. On the average, for each disambiguation query, SMML_{cov} took around 40 seconds (over 1M topics and 3M edges DAG) to infer the topics, whereas $\text{SMML}_{\text{cov+sim}}$ took around 35 minutes. Both these experiments were carried on a machine with 32 GB RAM, Eight-Core AMD Opteron(tm) Processor 2427.

5 Conclusions

We investigated a problem of summarizing topics over a massive topic DAG such that the summary set of topics produced represents the objects in the collection. This representation is characterized through various classes of submodular (and monotone) functions that captured coverage, similarity, diversity, specificity, clarity, relevance and fidelity of the topics. Currently we assume that the number of topics K is given as an input to our algorithm. It would be an interesting future problem to estimate the value of K automatically in our setting. As future work, we also plan to extend our techniques to produce a hierarchical summary of topics and scale it across heterogeneous collection of objects (from different domains) to bring all of them under the same topic DAG and investigate interesting cases thereon.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1162606, and by a Google, a Microsoft, and an Intel research award. Rishabh Iyer acknowledges support from the Microsoft Research Ph.D Fellowship.

References

Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. 2006. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, April.

W. Bi and J. T. Kwok. 2011. Multi-label classification on tree-and DAG-structured hierarchies. In *ICML*. ICML.

David M. Blei and John D. Lafferty. 2006. Correlated topic models. In *In Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. MIT Press.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, page 2003. MIT Press.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of Principles of Database Systems*, pages 155–166. ACM.

Razvan Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06), Trento, Italy*, pages 9–16, April.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. 2003. Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 178–186, New York, NY, USA. ACM.

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA.

R. Iyer, S. Jegelka, and J. Bilmes. 2013. Fast semidifferential-based submodular function optimization. ICML.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, Angela Y. Wu, Senior Member, and Senior Member. 2002. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:881–892.

D. Kempe, J. Kleinberg, and E. Tardos. 2003. Maximizing the spread of influence through a social network. In *SIGKDD*.

Katrin Kirchhoff and Jeff Bilmes. 2014. Submodularity for data selection in machine translation. October.

A. Krause and C. Guestrin. 2005. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of Uncertainity in Artificial Intelligence*. UAI.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 577–584, New York, NY, USA. ACM.

H. Lin and J. Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL*.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *The 49th Meeting of the Assoc. for Comp. Ling. Human Lang. Technologies (ACL/HLT-2011)*, Portland, OR, June.

H. Lin and J. Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, page 479490.

Hui Lin. 2012. *Submodularity in Natural Language Processing: Algorithms and Applications*. Ph.D. thesis, University of Washington, Dept. of EE.

Min ling Zhang and Zhi hua Zhou. 2007. Ml-knn: A lazy learning approach to multi-label learning. *PATTERN RECOGNITION*, 40:2007.

Arun S. Maiya, John P. Thompson, Francisco Loaiza-Lemos, and Robert M. Rolfe. 2013. Exploratory analysis of highly heterogeneous document collections. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1375–1383, New York, NY, USA. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Olena Medelyan, Ian H. Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the Wikipedia and AI workshop at AAAI-08*. AAAI.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 490–499, New York, NY, USA. ACM.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA. ACM.

David Milne. 2009. An open-source toolkit for mining wikipedia. In *In Proc. New Zealand Computer Science Research Student Conf*, page 2009.

Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, chapter 27, pages 234–243. Springer Berlin Heidelberg, Berlin/Heidelberg.

Mukund Narasimhan and Jeff Bilmes. 2004. PAC-learning bounded tree-width graphical models. In *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference (UAI-2004)*. Morgan Kaufmann Publishers, July.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294.

Adler J. Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. 2011. Hierarchically supervised latent dirichlet allocation. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *NIPS*, pages 2609–2617.

Juho Rousu, Craig Saunders, Sndor Szedmk, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626.

Jr. Silla, CarlosN. and AlexA. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Sebastian Tschiatschek, Rishabh Iyer, Hoachen Wei, and Jeff Bilmes. 2014. Learning Mixtures of Submodular Functions for Image Collection Summarization. In *Neural Information Processing Systems (NIPS)*.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer US.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2014a. Fast multi-stage submodular maximization. In *ICML*.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, Chris Bartels, and Jeff Bilmes. 2014b. Submodular subset selection for large-scale speech training data. *Proceedings of ICASSP, Florence, Italy*.

Torsten Zesch and Iryna Gurevych. 2007. Analysis of the wikipedia category graph for nlp applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, pages 1–8, Rochester, April. Association for Computational Linguistics.

Summarization of Multi-Document Topic Hierarchies using Submodular Mixtures

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, 40076, India
bairi@cse.iitb.ac.in

Rishabh Iyer
University of Washington
Seattle, WA-98175, USA
rkiyer@u.washington.edu

Ganesh Ramakrishnan
IIT Bombay
Mumbai, 40076, India
ganesh@cse.iitb.ac.in

Jeff Bilmes
University of Washington
Seattle, WA-98175, USA
bilmes@uw.edu

Abstract

We study the problem of summarizing DAG-structured topic hierarchies over a given set of documents. Example applications include automatically generating Wikipedia disambiguation pages for a set of articles, and generating candidate multi-labels for preparing machine learning datasets (e.g., for text classification, functional genomics, and image classification). Unlike previous work, which focuses on clustering the set of documents using the topic hierarchy as features, we directly pose the problem as a submodular optimization problem on a topic hierarchy using the documents as features. Desirable properties of the chosen topics include document coverage, specificity, topic diversity, and topic homogeneity, each of which, we show, is naturally modeled by a submodular function. Other information, provided say by unsupervised approaches such as LDA and its variants, can also be utilized by defining a submodular function that expresses coherence between the chosen topics and this information. We use a large-margin framework to learn convex mixtures over the set of submodular components. We empirically evaluate our method on the problem of automatically generating Wikipedia disambiguation pages using human generated clusterings as ground truth. We find that our framework improves upon several baselines according to a variety of standard evaluation metrics including the Jaccard Index, F1 score and NMI, and moreover, can be scaled to extremely large scale problems.

1 Introduction

Several real world machine learning applications involve hierarchy based categorization of topics for a set of objects. Objects could be, e.g., a set of documents for text classification, a set of genes in functional genomics, or a set of images in computer vision. One can often define a natural topic hierarchy to categorize these objects. For example, in text and image classification problems, each document or image is assigned a hierarchy of labels — a baseball page is assigned the labels "baseball" and "sports." Moreover, many of these applications, naturally have an existing topic hierarchy generated on the entire set of objects (Rousu et al., 2006; Barutcuoglu et al., 2006; ling Zhang and hua Zhou, 2007; Silla and Freitas, 2011; Tsoumakas et al., 2010).

Given a DAG-structured topic hierarchy and a subset of objects, we investigate the problem of finding a subset of DAG-structured topics that are induced by that subset (of objects). This problem arises naturally in several real world applications. For example, consider the problem of identifying appropriate label sets for a collection of articles. Several existing text collection datasets such as 20 Newsgroup[1], Reuters-21578[2] work with a predefined set of topics. We observe that these topic names are highly abstract[3] for the articles categorized under them. On the other hand, techniques proposed by systems such as Wikipedia Miner (Milne, 2009) and TAGME (Ferragina and Scaiella, 2010) generate several labels for each article in the dataset that are highly specific to the article. Collating all labels from all articles to create a label

[1] http://qwone.com/~jason/20Newsgroups/

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[3] Topic *Concept* is more abstract than the topic *Science* which is more abstract than the topic*Chemistry*

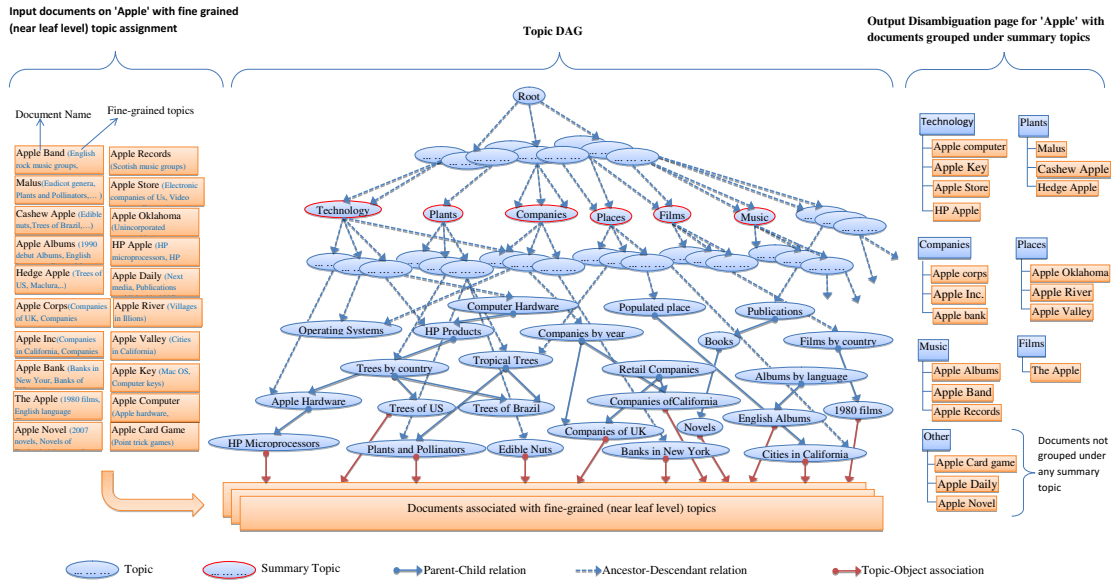

Figure 1: Topic Summarization overview. On the left, we show many documents related to Apple. In the middle, a Wikipedia category hierarchy shown as a topic DAG, links these documents at the leaf level. On the right, we show the output of our summarization process, which creates a set of summary topics (Plants, Technology, Companies, Films, Music and Places in this example) with the input documents classified under them.

set for the dataset can result in a large number of labels and become unmanageable. Our proposed techniques can summarize such large sets of labels into a smaller and more meaningful label sets using a DAG-structured topic hierarchy. This also holds for image classification problems and datasets like ImageNet (Deng et al., 2009). We use the term *summarize* to highlight the fact that the smaller label set semantically covers the larger label set. For example, the topics *Physics*, *Chemistry*, and *Mathematics* can be summarized into a topic *Science*.

A particularly important application of our work (and the one we use for our evaluations in Section 4) is the following: Given a collection of articles spanning different topics, but with similar titles, automatically generate a disambiguation page for those titles using the Wikipedia category hierarchy[4] as a topic DAG. Disambiguation pages[5] on Wikipedia are used to resolve conflicts in article titles that occur when a title is naturally associated with multiple articles on distinct topics. Each disambiguation page organizes articles into several groups, where the articles in each group pertain only to a specific topic. Disambiguations may be seen as paths in a hierarchy leading to different articles that arguably could have the same title. For example, the title *Apple*[6] can refer to a plant, a company, a film, a television show, a place, a technology, an album, a record label, and a newspaper daily. The problem then, is to organize the articles into multiple groups where each group contains articles of similar nature (topics) and has an appropriately discerned group heading. Figure 1 describes the topic summarization process for creation of the disambiguation page for "Apple".

All the above mentioned problems can be modeled as the problem of finding the most representative subset of topic nodes from a DAG-Structured topic hierarchy. We argue that many formulations of this problem are natural instances of submodular maximization, and provide a learning framework to create submodular mixtures to solve this problem. A set function $\mathrm{f}(.)$ is said to be submodular if for any element v and sets $A \subseteq B \subseteq V \setminus \{v\}$, where V represents the ground set of elements, $\mathrm{f}(A \cup \{v\}) - \mathrm{f}(A) \geq \mathrm{f}(B \cup \{v\}) - \mathrm{f}(B)$. This is called the diminishing returns property and states, informally, that adding an element to a smaller set increases the function value more than adding that element to a larger set. Submodular functions naturally model notions of coverage and diversity in applications, and therefore, a number of machine learning problems can be modeled as forms of submodular optimization (Kempe et al., 2003; Krause and Guestrin, 2005; Narasimhan and Bilmes, 2004; Iyer et al., 2013; Lin and Bilmes, 2012; Lin and Bilmes, 2010). In this paper, we investigate structured prediction methods for learn-

[4] http://en.wikipedia.org/wiki/Help:Categories

[5] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation

[6] http://en.wikipedia.org/wiki/Apple_(disambiguation)

ing weighted mixtures of submodular functions to summarize topics for a collection of objects using DAG-structured topic hierarchies. Throughout this paper we use the terms "topic" and "category" interchangeably.

1.1 Related Work

To the best of our knowledge, the specific problem we consider here is new. Previous work on identifying topics can be broadly categorized into one of the following types: a) cluster the objects and then identify names for the clusters; or b) dynamically identify topics (including hierarchical) for a set of objects. LDA (Blei et al., 2003) clusters the documents and simultaneously produces a set of topics into which the documents are clustered. In LDA, each document may be viewed as a mixture of various topics and the topic distribution is assumed to have a Dirichlet prior. LDA associates a group of high probability words to each identified topic. A name can be assigned to a topic by manually inspecting the words or using additional algorithms like (Mei et al., 2007; Maiya et al., 2013). LDA does not make use of existing topic hierarchies and correlation between topics. The Correlated Topic Model (Blei and Lafferty, 2006) induces a correlation structure between topics by using the logistic normal distribution instead of the Dirichlet. Another extension is the hierarchical LDA (Blei et al., 2004), where topics are joined together in a hierarchy by using the nested Chinese restaurant process. Nonparametric extensions of LDA include the Hierarchical Dirichlet Process (Teh et al., 2006) mixture model, which allows the number of topics to be unbounded and learnt from data and the Nested Chinese Restaurant Process which allows topics to be arranged in a hierarchy whose structure is learnt from data. In each of these approaches, unlike our proposed approach, an existing topic hierarchy is not used, nor is any additional object-topic information leveraged.

The pachinko allocation model (PAM)(Li and McCallum, 2006) captures arbitrary, nested, and possibly sparse correlations between topics using a DAG. The leaves of the DAG represent individual words in the vocabulary, while each interior node represents a correlation among its children, which may be words or other interior nodes (topics). PAM learns the probability distributions of words in a topic, subtopics in a topic, and topics in a document. We cannot, however, generate a subset of topics from a large existing topic DAG that can act as summary topics, using PAM.

HSLDA (Perotte et al., 2011) introduces a hierarchically supervised LDA model to infer hierarchical labels for a document. It assumes an existing label hierarchy in the form of a tree. The model infers one or more labels such that, if a label l is inferred as relevant to a document, then all the labels from l to the root of the tree are also inferred as relevant to the document. Our approach differs from HSLDA since: (1) we use the label hierarchy to infer a set of labels for a *group* of documents; (2) we do not enforce the label hierarchy to be a tree as it can be a DAG; and (3) generalizing HSLDA to use a DAG structured hierarchy and infer labels for a *group* of documents (e.g., combining into one big document) also may not help in solving our problem. HSLDA will apply *all* the relevant labels to the documents as per the classifier that it learns for every label. Moreover, the "root" label is always applied and it is very likely that many labels near the top level of the label hierarchy are also classified as relevant to the group of documents.

Wei and James (Bi and Kwok, 2011) present a hierarchical multi-label classification algorithm that can be used on both tree and DAG structured hierarchies. They formulate a search for the optimal consistent multi-label as the finding of the best subgraph in a tree/DAG. In our approach, we assume, individual documents are already associated with one or more topics and we find a consistent label set for a group of documents using the DAG structured topic hierarchy.

Medelyan et al. (Medelyan et al., 2008) and Ferragina et al. (Ferragina and Scaiella, 2010) detect topics for a document using Wikipedia article names and category names as the topic vocabulary. These systems are able to extract signals from a text document and identify Wikipedia articles and/or categories that optimally match the document and assign those article/category names as topics for the document. When run on a large collection of documents, these approaches generate enormous numbers of topics, a problem our proposed approach addresses.

1.2 Our Contributions

While most prior work discussed above focuses on the underlying set of documents, (*e.g.*, by clustering documents), we focus directly on the topics. In particular, we formulate the problem as subset selection on the set of topics within a DAG while simultaneously considering the documents to be categorized. Our method can scale to the colossal size of the DAG (1 million topics and 3 million correlation links between topics in Wikipedia). Moreover, our approach can

naturally incorporate outputs from many of the aforementioned algorithms. Our approach is based on submodular maximization and mixture learning, which has been successfully used in applications such as document summarization (Lin, 2012) and image summarization (Tschiatschek et al., 2014), but has never been applied to topic identification tasks or, more generally, DAG summarization.

We introduce a family of submodular functions to identify an appropriate set of topics from a DAG structured hierarchy of topics for a group of documents. We characterize this topic appropriateness through a set of desirable properties such as coverage, diversity, specificity, clarity, and relevance. Each of the submodular function components we consider are monotone, thereby ensuring a near optimal performance obtainable via a simple greedy algorithm for optimization.[7]. We also show how our technique naturally embodies outputs of other algorithms such as LDA, clustering, and classifications. Finally, we utilize a large margin formulation for learning mixtures of these submodular functions, and show how we can optimally learn them from training data.

Our approach demonstrates how to utilize the features collectively in the document space and the topic space to infer a set of topics. From an empirical perspective, we introduce and evaluate our approach on a dataset of around 8000 disambiguations that was extracted from Wikipedia and subsequently cleaned using the methods described in the experimentation section. We show that our learning framework outperforms many of the baselines, and is practical enough to be used on large corpora.

2 Problem Formulation

Let $G\,(V, E)$ be the DAG structured topic hierarchy with V topics. These topics are observed to have a parent child (*isa*) relationship forming a DAG. Let D be the set of documents that are associated with one or more of these topics. The middle portion of Figure 1 depicts a topic hierarchy with associated documents. The association links between the documents and topics can be hard or soft. In case of a hard link, a document is attached to a set of topics. Examples include multi-labeled documents. In case of a soft link, a document is associated with a topic with some degree of confidence (or probability). Furthermore, if a document is attached to a topic t, we assume that all the ancestor topics of t are also relevant for that document. This assumption has been employed in earlier works (Blei et al., 2004; Bi and Kwok, 2011; Rousu et al., 2006) as well. Given a budget of K, our objective is to choose a set of K topics from V, which best describe the documents in D. The notion of best describing topics is characterized through a set of desirable properties - coverage, diversity, specificity, clarity, relevance and fidelity - that K topics have to satisfy. The submodular functions that we introduce in the next section ensure these properties are satisfied. Formally, we solve the following discrete optimization problem:

$$S^* \in \underset{S \subseteq V : |S| \leq K}{\operatorname{argmax}} \sum_i w_i f_i(S) \qquad (1)$$

where, f_i are monotone submodular mixture components and $w_i \geq 0$ are the weights associated with those mixture components. Set S^* is the summary topics scored best.

It is easy to find massive (i.e., size in the order of million) DAG structured topic hierarchies in practice. Wikipedia's category hierarchy consists of more than 1M categories (topics) arranged hierarchically. In fact, they form a cyclic graph (Zesch and Gurevych, 2007). However, we can convert it to a DAG by eliminating the cycles as described in the supplementary material. YAGO (Suchanek et al., 2007) and Freebase (Bollacker et al., 2008) are other instances of massive topic hierarchies. The association of the documents with the existing topic hierarchy is also well studied. Systems such as WikipediaMiner (Milne, 2009), TAGME (Ferragina and Scaiella, 2010) and several annotation systems such as (Dill et al., 2003; Mihalcea and Csomai, 2007; Bunescu and Pasca, 2006) attach topics from Wikipedia (and other catalogs) to the documents by establishing the hard or soft links mentioned above.

Our goal is the following: Given a (ground set) collection V of topics organized in a pre-existing hierarchical DAG structure, and a collection D of documents, chose a size $K \in \mathbb{Z}_+$ representative subset of topics. Our approach is distinct from earlier work (e.g., (Kanungo et al., 2002; Blei et al., 2003)) where typically only a set of documents is classified and categorized in some way. We next provide a few definitions needed later in the paper.

Definition 1: Transitive Cover Γ): A topic t is said to cover a set of documents $\Gamma(t)$, called the transitive cover of the topic t, if for all documents $i \in \Gamma(t)$, either i is associated directly with topic t or with any of the descendant topics of t in the topic DAG. A natural extension of this definition to a set of topics T is defined as $\Gamma(T) = \cup_{t \in T} \Gamma(t)$.

[7] A simple greedy algorithm (Nemhauser et al., 1978) obtains a $1 - 1/e$ approximation guarantee for monotone submodular function maximization

Definition 2: Truncated Transitive Cover (Γ^α): This is a transitive cover of topic t, but with the limitation that the path length between a document and the topic t is not more than α. Hence, $|\Gamma^\alpha(t)| \leq |\Gamma(t)|$.

While our problem is closely related to clustering approaches, which consider the set of documents directly, there are some crucial differences. In particular, we focus on producing a clustering of documents where clusters are encouraged to honor a pre-defined DAG structured topic hierarchy. Existing agglomerative clustering algorithms focusing on the coverage of documents may not produce the desired clustering. To understand this, consider six documents d1, d2 ... d6 to be grouped into three clusters. There may be multiple ways to do this depending upon multiple aggregation paths present in the topic DAG: ((d1, d2), (d3, d4), (d5, d6)) or ((d1, d2, d3), (d4, d5), (d6)) or ((d1, d2, d3, d4), (d5), (d6)) or something else. Hence, we need more stringent measures to prefer one clustering over the others. Our work addresses this with a variety of quality criteria (coverage, diversity, specificity, clarity, relevance and fidelity, which are explained later in this paper) that are organically derived from well established submodular functions. And, most importantly, we learn the right mixture of these qualities to be enforced from the data itself. Furthermore, our approach also generalizes these clustering approaches, since one of the components in our mixture of submodular functions is defined via these unsupervised approaches, and maps a given clustering to a set of topics in the hierarchy.

3 Submodular Components and Learning

Summarization is the task of extracting information from a source that is both small in size but still representative. Our problem is different from traditional summarization tasks since we have an underlying DAG as a topic hierarchy that we wish to summarize in response to a subset of documents. Thus, a critical part of our problem is to take the graph structure into account while creating the summaries. Below, we identify properties we wish our summaries to posses.

Coverage: A summary set of topics should cover most of the documents. A document is said to be covered by a topic if there exists a path from the topic, going through intermediary descendant topics, to the document, i.e., the document is within the transitive cover of the topic.

Diversity: Summaries should be as diverse as possible, i.e., each summary topic should cover a unique set of documents. When a document is covered by more than one topic, that document is redundantly covered, e.g., "Finance" and "Banking" would be unlikely members of the same summary.

Summary qualities also involve "quality" notions, including:

Specificity/Clarity/Relevance/Coherence: These quality measures help us choose a set of topics that are neither too abstract nor overly specific. They ensure that the topics are clear and relevant to the documents that they represent. When additional information such as clustering (from LDA or other sources) and tagging (manual) documents is available, these quality criteria encourage the chosen topics to show resemblance (coherence) to those clustering/tagging in terms of transitive cover of documents they produce.

In the below, we define a variety of submodular functions that capture the above properties, and we then describe a large margin learning framework for learning convex mixtures of such components.

3.1 Submodular Components

3.1.1 Coverage Based Functions

Coverage components capture "coverage" of a set of documents.

Weighted Set Cover Function: Given a set of categories, $S \subseteq V$, define $\Gamma(S)$ as the set of documents covered — for each topic $s \in S$, $\Gamma(s) \subseteq D$ represents the documents covered by topic s and $\Gamma(S) = \cup_{s \in S} \Gamma(s)$. The weighted set cover function, defined as $f(S) = \sum_{d \in \Gamma(S)} w_d = w(\Gamma(S))$, assigns weights to the documents based on their relative importance (e.g., in Wikipedia disambiguation, the different documents could be ranked based on their priority).

Feature-based Functions: This class of function represents coverage in feature space. Given a set of categories $S \subseteq V$, and a set of features U, define $m_u(S)$ as the score associated with the set of categories S for feature $u \in U$. The feature set could represent, for example, the documents, in which case $m_u(S)$ represents the number of times document u is covered by the set S. U could also represent more complicated features. For example, in the context of Wikipedia disambiguation, U could represent TFIDF features over the documents. Feature based are then defined as $f(S) = \sum_{u \in U} \psi(m_u(S))$, where ψ is a concave (e.g., the square root) function. This function class has been successfully used in several applications (Kirchhoff and Bilmes, 2014; Wei et al., 2014a; Wei et al., 2014b).

3.1.2 Similarity based Functions

Similarity functions are defined through a similarity matrix $\mathbf{S} = \{s_{ij}\}_{i,j \in V}$. Given categories $i, j \in V$, similarity s_{ij} in our case can be defined as $s_{ij} = |\Gamma(i) \cap \Gamma(j)|$, i.e the number of documents commonly covered by both i and j.

Facility Location: The facility location function, defined as $f(S) = \sum_{i \in V} \max_{j \in S} s_{ij}$, is a natural model for k-medoids and exemplar based clustering, and has been used in several summarization problems (Tschiatschek et al., 2014; Wei et al., 2014a).

Penalty based diversity: A similarity matrix may be used to express a form of coverage of a set S but that is then penalized with a redundancy term, as in the following difference: $f(S) = \sum_{i \in V, j \in S} s_{ij} - \lambda \sum_{i \in S} \sum_{j \in S,} s_{i,j}$ (Lin and Bilmes, 2011)). Here $\lambda \in [0, 1]$. This function is submodular, but is not in general monotone, and has been used in document summarization (Lin and Bilmes, 2011), as a dispersion function (Borodin et al., 2012), and in image summarization (Tschiatschek et al., 2014).

3.1.3 Quality Control (QC) Functions

QC functions ensure a quality criteria is met by a set S of topics. We define the quality score of the set S as $F_q(S) = \sum_{s \in S} f_q(s)$, where $f_q(s)$ is the quality score of topic s for quality q. Therefore, $F_q(S)$ is a modular function in S. We investigate three types of quality control functions: Topic Specificity, Topic Clarity, and Topic Relevance.

Topic Specificity: The farther a topic is from the root of the DAG, the more specific it becomes. Topics higher up in the hierarchy are abstract and less specific. We therefore prefer topics low in the DAG, but lower topics also have less coverage. We define $f_{\text{specificity}}(s) = s_h$ where s_h is the height of topic s in the DAG. The root topic has height zero and the "height" increases as we move down the DAG in Figure 1.

Topic Clarity: Topic clarity is the fraction of descendant topics that cover one or more documents. If a topic has many descendant topics that do not cover any documents, it has less clarity. Formally, $f_{\text{clarity}}(s) = \frac{\sum_{t \in \text{descendants}(s)} [\![\Gamma(t) > 0]\!]}{|\text{descendants}(s)|}$, where $[\![.]\!]$ is the indicator function.

Topic Relevance: A topic is considered to be better related to a document if the number of hops needed to reach the document from that topic is lower. Given any set $A \subseteq D$ of document, and any topic $s \in V$, we can define $f_{\text{relevance}}(s|A) = \text{argmin}_\alpha \{\alpha : A \subseteq \Gamma^\alpha(s)\}$.

QC Functions As Barrier Modular Mixtures: We introduce a modular function for every QC function as follows

$$f^\alpha_{\text{specificity}}(s) = \begin{cases} 1 & \text{if the height of topic } s \text{ is at least } \alpha \\ 0 & \text{otherwise} \end{cases}$$

for every possible value of α. This creates a submodular mixture with as many components as the number of possible values of α. In our experiments with Wikipedia, we had α varying from 1 to 120 stepping by 1, adding 120 modular mixture components. Similarly, we define,

$$f^\beta_{\text{clarity}}(s) = \begin{cases} 1 & \text{if the clarity of topic } s \text{ is at least } \beta \\ 0 & \text{otherwise} \end{cases}$$

for every possible (discretized to make it countably finite) value of β. And,

$f^\gamma_{\text{relevance}}(s) = f_{\text{cov}}(s|\Gamma^\gamma(s))$, where $f_{\text{cov}}(.)$ is the coverage submodular function and $s|X$ indicates coverage of a topic s over a set of documents X. All these functions (modular and submodular terms) are added as mixture components in our learning framework to learn suitable weights for them. We then use these weights in our inference procedure to obtain a subset of topics as described in 3.2. We show from our experiments that this approach performs better than all other approaches and baselines.

3.1.4 Fidelity Functions

A function representing the fidelity of a set S to another reference set R is one that gets a large value when the set S represents the set R. Such a function scores inferred topics high when it resembles a reference set of topics and/or item clusters. The reference set in this case can be produced from other algorithms such as k-means, LDA and its variants or from a manually tagged corpus. Next we describe one such fidelity function.

Topic Coherence: This function scores a set of topics S high when the transitive cover (Definition 1) produced by the topics in S resembles the clusters of documents produced by an external source (k-means, LDA or manual). Given an external source that clusters the documents, producing T clusters $L_1, L_2, ..., L_T$ (for T topics), topic coherence is defined as: $f(S) = \sum_{t \in T} \max_{k \in S} w_{k,t}$ where $w_{k,t} = \text{harmonic\_mean}(w^p_{k,t}, w^r_{k,t})$ and $w^p_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|\Gamma(k)|}$ and $w^r_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|L_t|}$. Note that, $w^p_{k,t} \geq 0$ and $w^r_{k,t} \geq 0$ are the precision on recall of the resemblance and $w_{k,t}$ is the F1 measure. If the transitive cover of topics in S resembles the reference clusters L_t exactly, we attain maximum coherence (or fidelity). As the resemblance diminishes, the score decreases. The above function $f(S)$ is monotone submodular.

3.1.5 Mixture of Submodular Components:

Given the different classes of submodular functions above, we construct our submodular scoring functions $F_w(\cdot)$ as a convex combinations of these different submodular functions $f_1, f_2, \ldots, f_m$, above. In other words,

$$F_w(S) = \sum_{i=1}^{m} w_i f_i(S), \tag{2}$$

where $w = (w_1, \ldots, w_m)$, $w_i \geq 0, \sum_i w_i = 1$. The components f_i are submodular and assumed to be normalized: i.e., $f_i(\emptyset) = 0$, and $f_i(V) = 1$ for monotone functions and $\max_{A \subseteq V} f_i(A) \leq 1$ for non-monotone functions. A simple way to normalize a monotone submodular function is to define the component as $f_i(S)/f_i(V)$. This ensures that the components are *compatible* with each other. Obviously, the merit of the scoring function $F_w(\cdot)$ depends on the selection of the components.

3.2 Large Margin Learning

We optimize the weights w of the scoring function $F_w(\cdot)$ in a large-margin structured prediction framework. In this setting, we assume we have training data in the form of pairs of a set of documents, and a human generated summary as a set of topics. For example, in the case of Wikipedia disambiguation, we use the human generated disambiguation pages as the ground truth summary. We represent the set of ground-truth summaries as $\mathcal{S} = \{S_1, S_2, \cdots, S_N\}$. In large margin training, the weights are optimized such that ground-truth summaries $\mathcal{S}$ are separated from competitor summaries by a loss-dependent margin:

$$F_w(S) \geq F_w(S') + \mathcal{L}(S'), \quad \forall S \in \mathcal{S}, S' \in \mathcal{Y} \setminus \mathcal{S}, \tag{3}$$

where $\mathcal{L}(\cdot)$ is the loss function, and where $\mathcal{Y}$ is a structured output space (for example $\mathcal{Y}$ is the set of summaries that satisfy a certain budget B, i.e., $\mathcal{Y} = \{S' \subseteq V : |S'| \leq B\}$). We assume the loss to be normalized, $0 \leq \mathcal{L}(S') \leq 1, \forall S' \subseteq V$, to ensure that mixture and loss are calibrated. Equation (3) can be stated as $F_w(S) \geq \max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right], \forall S \in \mathcal{S}$ which is called *loss-augmented inference*. We introduce slack variables and minimize the regularized sum of slacks (Lin and Bilmes, 2012):

$$\min_{w \geq 0, \|w\|_1 = 1} \quad \sum_{S \in \mathcal{S}} \left[\max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right] - F_w(S)\right] + \frac{\lambda}{2}\|w\|_2^2, \tag{4}$$

where the non-negative orthant constraint, $w \geq 0$, ensures that the final mixture is submodular. Note a 2-norm regularizer is used on top of a 1-norm constraint $\|w\|_1 = 1$ which we interpret as a prior to encourage higher entropy, and thus more diverse mixture distributions. Tractability depends on the choice of the loss function. The parameters w are learnt using stochastic gradient descent as in (Tschiatschek et al., 2014).

3.3 Loss Functions

A natural choice of loss functions for our case can be derived from cluster evaluation metrics. Every inferred topic s induces a subset of documents, namely the transitive cover $\Gamma(s)$ of s. We compare these clusters with the clusters induced from the true topics in the training set and compute the loss.

In this paper, we use the Jaccard Index (JI) as a loss function. Let S be the inferred topics and T be the true topics. The Jaccard loss is defined as $\mathcal{L}_{\text{jaccard}}(S,T) = 1 - \frac{1}{k}\sum_{s \in S} \max_{t \in T} \frac{|\Gamma(s) \cap \Gamma(t)|}{|\Gamma(s) \cup \Gamma(t)|}$, where $k = |S| = |T|$ is the number of topics. When the clustering produced by the inferred and the true topics are similar, Jaccard loss is 0. When they are completely dissimilar, the loss is maximum, i.e., 1. Jaccard loss is a modular function.

3.4 Inference Algorithm: Greedy

Having learnt the weights for the mixture components, the resulting function $F_w(S) = \sum_{i=1}^{m} w_i f_i(S)$ is a submodular function. In the case when the individual components are themselves monotone (all our functions in fact are), $F_w(S)$ can be optimized by the accelerated greedy algorithm (Minoux, 1978). Thanks to submodularity, we can obtain near optimal solutions very efficiently. In case the functions are all monotone submodular, we can guarantee that the solution is within $1 - 1/e$ factor from the optimal solution.

4 Experimental Results

To validate our approach, we make use of Wikipedia category structure as a topic DAG and apply our technique to the task of automatic generation of Wikipedia disambiguation pages. We pre-processed the category graph to eliminate the cycles in order to make it a DAG. Each Wikipedia disambiguation page is manually created by Wikipedia editors by grouping a collection of Wikipedia articles into several groups. Each group is then assigned a name, which serves as a topic for the group. Typically, a disambiguation page segregates around 20-30 articles into 5-6 groups. Our goal is to measure how accurately we can recreate the groups for a disambiguation page and label them, given only the collection of articles mentioned in that disambiguation page (when actual groupings and labels are hidden).

4.1 Datasets

We parsed the contents of Wikipedia disambiguation pages and extracted disambiguation page names, article groups and group names. We collected about 8000 disambiguation pages that had at least four groups on them. Wikipedia category structure is used as the topic DAG. We eliminated few administrative categories such as "Hidden Categories", "Articles needing cleanup", and the like. The final DAG had about 1M topics and 3M links.

4.2 Evaluation Metrics

Every group of articles on the Wikipedia disambiguation page is assigned a name by the editors. Unfortunately, these names may not correspond to the Wikipedia category (topic) names. For example, one of the groups on the "Matrix" disambiguation page has a name "Business and government" and there is no Wikipedia category by that name. However, the group names generated by our (and baseline) method are from the Wikipedia categories (which forms our topic DAG). In addition, there can be multiple relevant names for a group. For example, a group on a disambiguation page may be called "Calculus", but an algorithm may rightly generate "Vector Calculus". Hence we cannot evaluate the accuracy of an algorithm just by matching the generated group names to those on the disambiguation page. To alleviate this problem, we adopt cluster-based evaluation metrics. We treat every group of articles generated by an algorithm under a topic for a disambiguation page as a cluster of articles. These are considered as *inferred* clusters for a disambiguation page. We compare them against the actual grouping of articles on the Wikipedia disambiguation page by treating those groups as *true* clusters. We can now adopt Jaccard Index, F1-measure, and NMI (Normalized Mutual Information) based cluster evaluation metrics described in (Manning et al., 2008). For each disambiguation page in the test set, we compute every metric score and then average it over all the disambiguation pages.

4.3 Methods Compared

We validated our approach by comparing against several baselines described below. We also compared two variations of our approach as described next. For each of these cases (baselines and two variations) we generated and compared the metrics (Jaccard Index, F1-measure and NMI) as described in the previous section.

KM_{docs}: K-Means algorithm run on articles as TF-IDF vectors of words. The number of clusters K is set to the number of true clusters on the Wikipedia disambiguation page.

$\text{KMed}_{\text{docs}}$: K-Medoids algorithm with articles as TF-IDF vectors of words. The number of clusters are set as in KM_{docs}.

$\text{KMed}_{\text{topics}}$: K-Medoids run on topics as TF-IDF vectors of words. The words for each topic is taken from the articles that are in the transitive cover of the topic.

LDA_{docs}: LDA algorithm with the number of topics set to the number of true clusters on the Wikipedia disambiguation page. Each article is then grouped under the highest probability topic.

SMML_{cov}: This is the submodular mixture learning case explained in section 3.1.5. Here we consider a mixture of all the submodular functions governing coverage, diversity, fidelity and QC functions. However, we exclude the similarity based functions described in section 3.1.2. Coverage based functions have a time complexity of $O(n)$ whereas similarity based functions are $O(n^2)$. By excluding similarity based functions, we can compare the quality of the results with and without $O(n^2)$ functions. We learn the mixture weights from the training set and use them during inference on the test set to subset K topics through the submodular maximization (Equation 1).

$\text{SMML}_{\text{cov+sim}}$: This case is similar to SMML_{cov} except that, we include similarity based submodular mixture components. This makes the inference time complexity $O(n^2)$.

We do not compare against HSLDA, PAM and few other techniques cited in the related work sections because they do not produce a subset of K summary topics — these are not directly comparable with our work.

4.4 Evaluation Results

We show that the submodular mixture learning and maximization approaches, i.e., SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other approaches in various metrics. In all these experiments, we performed 5 fold cross validation to learn the parameters from 80% of the disambiguation pages and evaluated on the rest of the 20%, in each fold.

In Figure 2a we summarize the results of the comparison of the methods mentioned above on Jaccard Index, F1 measure and NMI. Our proposed techniques SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other techniques consistently.

In Figures 2b and 2c we measure the number of test instances (i.e., disambiguation queries) in which each of the algorithms dominate (win) in evaluation metrics. In 60% of the disambiguation queries, SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ approaches

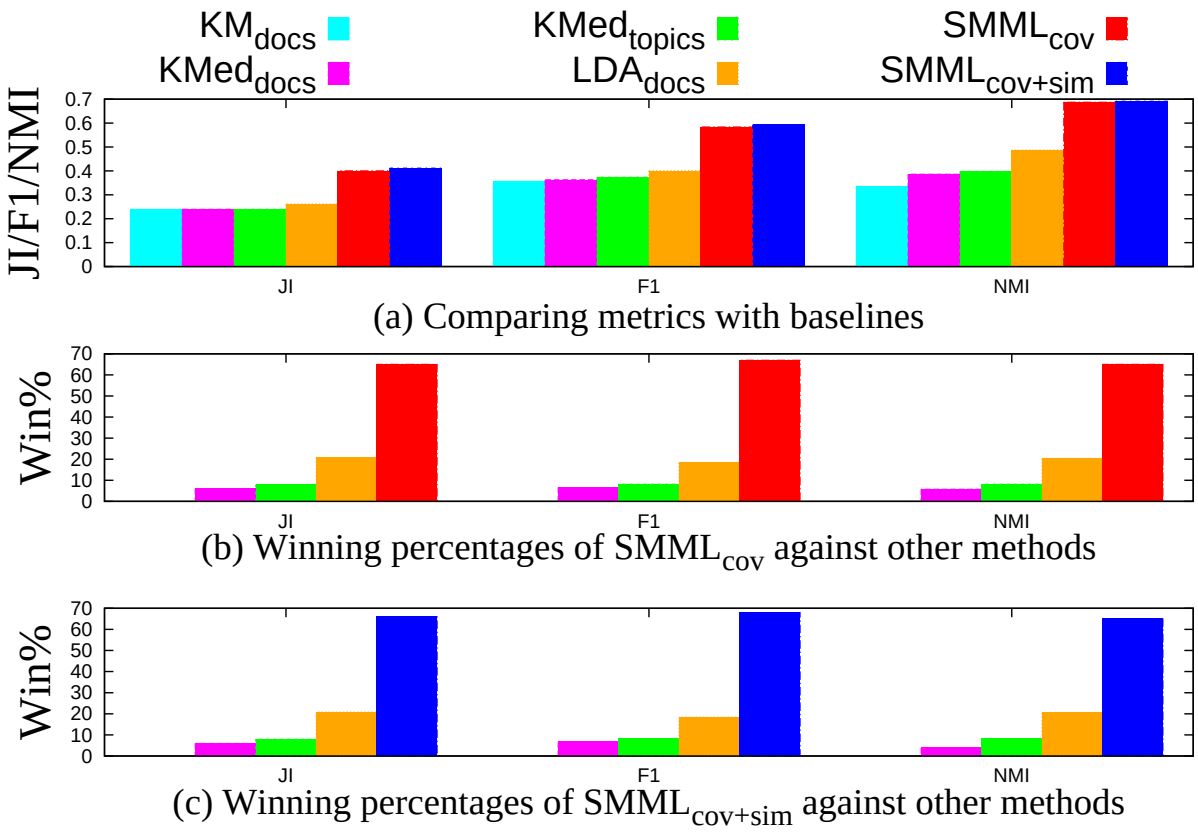

Figure 2: Comparison of techniques

produce higher JI, F1 and NMI than all other methods. This indicates that the clusters of articles produced by our technique resembles the clusters of articles present on the disambiguation page better than other techniques.

From Figures 2b and 2c it is clear that $O(n)$ time complexity based submodular mixture functions (SMML_{cov}) perform on par with $O(n^2)$ based functions ($\text{SMML}_{\text{cov+sim}}$), but at a greatly reduced execution time, demonstrating the sufficiency of $O(n)$ functions for our task. On the average, for each disambiguation query, SMML_{cov} took around 40 seconds (over 1M topics and 3M edges DAG) to infer the topics, whereas $\text{SMML}_{\text{cov+sim}}$ took around 35 minutes. Both these experiments were carried on a machine with 32 GB RAM, Eight-Core AMD Opteron(tm) Processor 2427.

5 Conclusions

We investigated a problem of summarizing topics over a massive topic DAG such that the summary set of topics produced represents the objects in the collection. This representation is characterized through various classes of submodular (and monotone) functions that captured coverage, similarity, diversity, specificity, clarity, relevance and fidelity of the topics. Currently we assume that the number of topics K is given as an input to our algorithm. It would be an interesting future problem to estimate the value of K automatically in our setting. As future work, we also plan to extend our techniques to produce a hierarchical summary of topics and scale it across heterogeneous collection of objects (from different domains) to bring all of them under the same topic DAG and investigate interesting cases thereon.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1162606, and by a Google, a Microsoft, and an Intel research award. Rishabh Iyer acknowledges support from the Microsoft Research Ph.D Fellowship.

References

Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. 2006. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, April.

W. Bi and J. T. Kwok. 2011. Multi-label classification on tree-and DAG-structured hierarchies. In *ICML*. ICML.

David M. Blei and John D. Lafferty. 2006. Correlated topic models. In *In Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. MIT Press.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, page 2003. MIT Press.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of Principles of Database Systems*, pages 155–166. ACM.

Razvan Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06), Trento, Italy*, pages 9–16, April.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. 2003. Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 178–186, New York, NY, USA. ACM.

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA.

R. Iyer, S. Jegelka, and J. Bilmes. 2013. Fast semidifferential-based submodular function optimization. ICML.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, Angela Y. Wu, Senior Member, and Senior Member. 2002. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:881–892.

D. Kempe, J. Kleinberg, and E. Tardos. 2003. Maximizing the spread of influence through a social network. In *SIGKDD*.

Katrin Kirchhoff and Jeff Bilmes. 2014. Submodularity for data selection in machine translation. October.

A. Krause and C. Guestrin. 2005. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of Uncertainity in Artificial Intelligence*. UAI.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 577–584, New York, NY, USA. ACM.

H. Lin and J. Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL*.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *The 49th Meeting of the Assoc. for Comp. Ling. Human Lang. Technologies (ACL/HLT-2011)*, Portland, OR, June.

H. Lin and J. Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, page 479490.

Hui Lin. 2012. *Submodularity in Natural Language Processing: Algorithms and Applications*. Ph.D. thesis, University of Washington, Dept. of EE.

Min ling Zhang and Zhi hua Zhou. 2007. Ml-knn: A lazy learning approach to multi-label learning. *PATTERN RECOGNITION*, 40:2007.

Arun S. Maiya, John P. Thompson, Francisco Loaiza-Lemos, and Robert M. Rolfe. 2013. Exploratory analysis of highly heterogeneous document collections. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1375–1383, New York, NY, USA. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Olena Medelyan, Ian H. Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the Wikipedia and AI workshop at AAAI-08*. AAAI.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 490–499, New York, NY, USA. ACM.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA. ACM.

David Milne. 2009. An open-source toolkit for mining wikipedia. In *In Proc. New Zealand Computer Science Research Student Conf*, page 2009.

Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, chapter 27, pages 234–243. Springer Berlin Heidelberg, Berlin/Heidelberg.

Mukund Narasimhan and Jeff Bilmes. 2004. PAC-learning bounded tree-width graphical models. In *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference (UAI-2004)*. Morgan Kaufmann Publishers, July.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294.

Adler J. Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. 2011. Hierarchically supervised latent dirichlet allocation. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *NIPS*, pages 2609–2617.

Juho Rousu, Craig Saunders, Sndor Szedmk, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626.

Jr. Silla, CarlosN. and AlexA. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Sebastian Tschiatschek, Rishabh Iyer, Hoachen Wei, and Jeff Bilmes. 2014. Learning Mixtures of Submodular Functions for Image Collection Summarization. In *Neural Information Processing Systems (NIPS)*.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer US.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2014a. Fast multi-stage submodular maximization. In *ICML*.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, Chris Bartels, and Jeff Bilmes. 2014b. Submodular subset selection for large-scale speech training data. *Proceedings of ICASSP, Florence, Italy*.

Torsten Zesch and Iryna Gurevych. 2007. Analysis of the wikipedia category graph for nlp applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, pages 1–8, Rochester, April. Association for Computational Linguistics.

Summarization of Multi-Document Topic Hierarchies using Submodular Mixtures

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, 40076, India
bairi@cse.iitb.ac.in

Rishabh Iyer
University of Washington
Seattle, WA-98175, USA
rkiyer@u.washington.edu

Ganesh Ramakrishnan
IIT Bombay
Mumbai, 40076, India
ganesh@cse.iitb.ac.in

Jeff Bilmes
University of Washington
Seattle, WA-98175, USA
bilmes@uw.edu

Abstract

We study the problem of summarizing DAG-structured topic hierarchies over a given set of documents. Example applications include automatically generating Wikipedia disambiguation pages for a set of articles, and generating candidate multi-labels for preparing machine learning datasets (e.g., for text classification, functional genomics, and image classification). Unlike previous work, which focuses on clustering the set of documents using the topic hierarchy as features, we directly pose the problem as a submodular optimization problem on a topic hierarchy using the documents as features. Desirable properties of the chosen topics include document coverage, specificity, topic diversity, and topic homogeneity, each of which, we show, is naturally modeled by a submodular function. Other information, provided say by unsupervised approaches such as LDA and its variants, can also be utilized by defining a submodular function that expresses coherence between the chosen topics and this information. We use a large-margin framework to learn convex mixtures over the set of submodular components. We empirically evaluate our method on the problem of automatically generating Wikipedia disambiguation pages using human generated clusterings as ground truth. We find that our framework improves upon several baselines according to a variety of standard evaluation metrics including the Jaccard Index, F1 score and NMI, and moreover, can be scaled to extremely large scale problems.

1 Introduction

Several real world machine learning applications involve hierarchy based categorization of topics for a set of objects. Objects could be, e.g., a set of documents for text classification, a set of genes in functional genomics, or a set of images in computer vision. One can often define a natural topic hierarchy to categorize these objects. For example, in text and image classification problems, each document or image is assigned a hierarchy of labels — a baseball page is assigned the labels "baseball" and "sports." Moreover, many of these applications, naturally have an existing topic hierarchy generated on the entire set of objects (Rousu et al., 2006; Barutcuoglu et al., 2006; ling Zhang and hua Zhou, 2007; Silla and Freitas, 2011; Tsoumakas et al., 2010).

Given a DAG-structured topic hierarchy and a subset of objects, we investigate the problem of finding a subset of DAG-structured topics that are induced by that subset (of objects). This problem arises naturally in several real world applications. For example, consider the problem of identifying appropriate label sets for a collection of articles. Several existing text collection datasets such as 20 Newsgroup[1], Reuters-21578[2] work with a predefined set of topics. We observe that these topic names are highly abstract[3] for the articles categorized under them. On the other hand, techniques proposed by systems such as Wikipedia Miner (Milne, 2009) and TAGME (Ferragina and Scaiella, 2010) generate several labels for each article in the dataset that are highly specific to the article. Collating all labels from all articles to create a label

[1] http://qwone.com/~jason/20Newsgroups/

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[3] Topic *Concept* is more abstract than the topic *Science* which is more abstract than the topic*Chemistry*

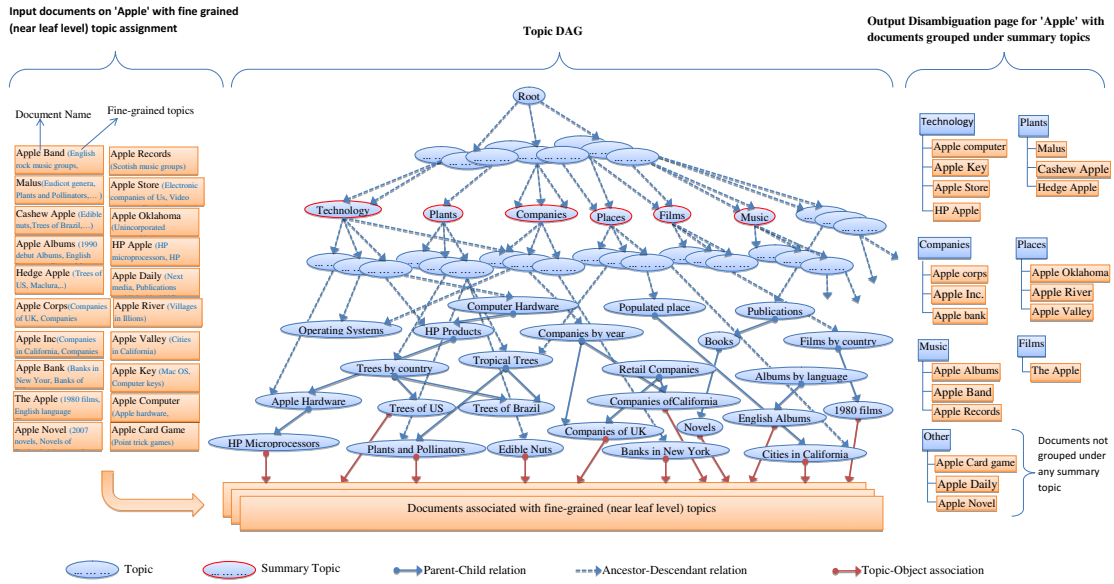

Figure 1: Topic Summarization overview. On the left, we show many documents related to Apple. In the middle, a Wikipedia category hierarchy shown as a topic DAG, links these documents at the leaf level. On the right, we show the output of our summarization process, which creates a set of summary topics (Plants, Technology, Companies, Films, Music and Places in this example) with the input documents classified under them.

set for the dataset can result in a large number of labels and become unmanageable. Our proposed techniques can summarize such large sets of labels into a smaller and more meaningful label sets using a DAG-structured topic hierarchy. This also holds for image classification problems and datasets like ImageNet (Deng et al., 2009). We use the term *summarize* to highlight the fact that the smaller label set semantically covers the larger label set. For example, the topics *Physics*, *Chemistry*, and *Mathematics* can be summarized into a topic *Science*.

A particularly important application of our work (and the one we use for our evaluations in Section 4) is the following: Given a collection of articles spanning different topics, but with similar titles, automatically generate a disambiguation page for those titles using the Wikipedia category hierarchy[4] as a topic DAG. Disambiguation pages[5] on Wikipedia are used to resolve conflicts in article titles that occur when a title is naturally associated with multiple articles on distinct topics. Each disambiguation page organizes articles into several groups, where the articles in each group pertain only to a specific topic. Disambiguations may be seen as paths in a hierarchy leading to different articles that arguably could have the same title. For example, the title *Apple*[6] can refer to a plant, a company, a film, a television show, a place, a technology, an album, a record label, and a newspaper daily. The problem then, is to organize the articles into multiple groups where each group contains articles of similar nature (topics) and has an appropriately discerned group heading. Figure 1 describes the topic summarization process for creation of the disambiguation page for "Apple".

All the above mentioned problems can be modeled as the problem of finding the most representative subset of topic nodes from a DAG-Structured topic hierarchy. We argue that many formulations of this problem are natural instances of submodular maximization, and provide a learning framework to create submodular mixtures to solve this problem. A set function $\mathrm{f}(.)$ is said to be submodular if for any element v and sets $A \subseteq B \subseteq V \setminus \{v\}$, where V represents the ground set of elements, $\mathrm{f}(A \cup \{v\}) - \mathrm{f}(A) \geq \mathrm{f}(B \cup \{v\}) - \mathrm{f}(B)$. This is called the diminishing returns property and states, informally, that adding an element to a smaller set increases the function value more than adding that element to a larger set. Submodular functions naturally model notions of coverage and diversity in applications, and therefore, a number of machine learning problems can be modeled as forms of submodular optimization (Kempe et al., 2003; Krause and Guestrin, 2005; Narasimhan and Bilmes, 2004; Iyer et al., 2013; Lin and Bilmes, 2012; Lin and Bilmes, 2010). In this paper, we investigate structured prediction methods for learn-

[4] http://en.wikipedia.org/wiki/Help:Categories

[5] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation

[6] http://en.wikipedia.org/wiki/Apple_(disambiguation)

ing weighted mixtures of submodular functions to summarize topics for a collection of objects using DAG-structured topic hierarchies. Throughout this paper we use the terms "topic" and "category" interchangeably.

1.1 Related Work

To the best of our knowledge, the specific problem we consider here is new. Previous work on identifying topics can be broadly categorized into one of the following types: a) cluster the objects and then identify names for the clusters; or b) dynamically identify topics (including hierarchical) for a set of objects. LDA (Blei et al., 2003) clusters the documents and simultaneously produces a set of topics into which the documents are clustered. In LDA, each document may be viewed as a mixture of various topics and the topic distribution is assumed to have a Dirichlet prior. LDA associates a group of high probability words to each identified topic. A name can be assigned to a topic by manually inspecting the words or using additional algorithms like (Mei et al., 2007; Maiya et al., 2013). LDA does not make use of existing topic hierarchies and correlation between topics. The Correlated Topic Model (Blei and Lafferty, 2006) induces a correlation structure between topics by using the logistic normal distribution instead of the Dirichlet. Another extension is the hierarchical LDA (Blei et al., 2004), where topics are joined together in a hierarchy by using the nested Chinese restaurant process. Nonparametric extensions of LDA include the Hierarchical Dirichlet Process (Teh et al., 2006) mixture model, which allows the number of topics to be unbounded and learnt from data and the Nested Chinese Restaurant Process which allows topics to be arranged in a hierarchy whose structure is learnt from data. In each of these approaches, unlike our proposed approach, an existing topic hierarchy is not used, nor is any additional object-topic information leveraged.

The pachinko allocation model (PAM)(Li and McCallum, 2006) captures arbitrary, nested, and possibly sparse correlations between topics using a DAG. The leaves of the DAG represent individual words in the vocabulary, while each interior node represents a correlation among its children, which may be words or other interior nodes (topics). PAM learns the probability distributions of words in a topic, subtopics in a topic, and topics in a document. We cannot, however, generate a subset of topics from a large existing topic DAG that can act as summary topics, using PAM.

HSLDA (Perotte et al., 2011) introduces a hierarchically supervised LDA model to infer hierarchical labels for a document. It assumes an existing label hierarchy in the form of a tree. The model infers one or more labels such that, if a label l is inferred as relevant to a document, then all the labels from l to the root of the tree are also inferred as relevant to the document. Our approach differs from HSLDA since: (1) we use the label hierarchy to infer a set of labels for a *group* of documents; (2) we do not enforce the label hierarchy to be a tree as it can be a DAG; and (3) generalizing HSLDA to use a DAG structured hierarchy and infer labels for a *group* of documents (e.g., combining into one big document) also may not help in solving our problem. HSLDA will apply *all* the relevant labels to the documents as per the classifier that it learns for every label. Moreover, the "root" label is always applied and it is very likely that many labels near the top level of the label hierarchy are also classified as relevant to the group of documents.

Wei and James (Bi and Kwok, 2011) present a hierarchical multi-label classification algorithm that can be used on both tree and DAG structured hierarchies. They formulate a search for the optimal consistent multi-label as the finding of the best subgraph in a tree/DAG. In our approach, we assume, individual documents are already associated with one or more topics and we find a consistent label set for a group of documents using the DAG structured topic hierarchy.

Medelyan et al. (Medelyan et al., 2008) and Ferragina et al. (Ferragina and Scaiella, 2010) detect topics for a document using Wikipedia article names and category names as the topic vocabulary. These systems are able to extract signals from a text document and identify Wikipedia articles and/or categories that optimally match the document and assign those article/category names as topics for the document. When run on a large collection of documents, these approaches generate enormous numbers of topics, a problem our proposed approach addresses.

1.2 Our Contributions

While most prior work discussed above focuses on the underlying set of documents, (*e.g.*, by clustering documents), we focus directly on the topics. In particular, we formulate the problem as subset selection on the set of topics within a DAG while simultaneously considering the documents to be categorized. Our method can scale to the colossal size of the DAG (1 million topics and 3 million correlation links between topics in Wikipedia). Moreover, our approach can

naturally incorporate outputs from many of the aforementioned algorithms. Our approach is based on submodular maximization and mixture learning, which has been successfully used in applications such as document summarization (Lin, 2012) and image summarization (Tschiatschek et al., 2014), but has never been applied to topic identification tasks or, more generally, DAG summarization.

We introduce a family of submodular functions to identify an appropriate set of topics from a DAG structured hierarchy of topics for a group of documents. We characterize this topic appropriateness through a set of desirable properties such as coverage, diversity, specificity, clarity, and relevance. Each of the submodular function components we consider are monotone, thereby ensuring a near optimal performance obtainable via a simple greedy algorithm for optimization.[7]. We also show how our technique naturally embodies outputs of other algorithms such as LDA, clustering, and classifications. Finally, we utilize a large margin formulation for learning mixtures of these submodular functions, and show how we can optimally learn them from training data.

Our approach demonstrates how to utilize the features collectively in the document space and the topic space to infer a set of topics. From an empirical perspective, we introduce and evaluate our approach on a dataset of around 8000 disambiguations that was extracted from Wikipedia and subsequently cleaned using the methods described in the experimentation section. We show that our learning framework outperforms many of the baselines, and is practical enough to be used on large corpora.

2 Problem Formulation

Let $G\,(V, E)$ be the DAG structured topic hierarchy with V topics. These topics are observed to have a parent child (*isa*) relationship forming a DAG. Let D be the set of documents that are associated with one or more of these topics. The middle portion of Figure 1 depicts a topic hierarchy with associated documents. The association links between the documents and topics can be hard or soft. In case of a hard link, a document is attached to a set of topics. Examples include multi-labeled documents. In case of a soft link, a document is associated with a topic with some degree of confidence (or probability). Furthermore, if a document is attached to a topic t, we assume that all the ancestor topics of t are also relevant for that document. This assumption has been employed in earlier works (Blei et al., 2004; Bi and Kwok, 2011; Rousu et al., 2006) as well. Given a budget of K, our objective is to choose a set of K topics from V, which best describe the documents in D. The notion of best describing topics is characterized through a set of desirable properties - coverage, diversity, specificity, clarity, relevance and fidelity - that K topics have to satisfy. The submodular functions that we introduce in the next section ensure these properties are satisfied. Formally, we solve the following discrete optimization problem:

$$S^* \in \underset{S \subseteq V : |S| \leq K}{\operatorname{argmax}} \sum_i w_i f_i(S) \qquad (1)$$

where, f_i are monotone submodular mixture components and $w_i \geq 0$ are the weights associated with those mixture components. Set S^* is the summary topics scored best.

It is easy to find massive (i.e., size in the order of million) DAG structured topic hierarchies in practice. Wikipedia's category hierarchy consists of more than 1M categories (topics) arranged hierarchically. In fact, they form a cyclic graph (Zesch and Gurevych, 2007). However, we can convert it to a DAG by eliminating the cycles as described in the supplementary material. YAGO (Suchanek et al., 2007) and Freebase (Bollacker et al., 2008) are other instances of massive topic hierarchies. The association of the documents with the existing topic hierarchy is also well studied. Systems such as WikipediaMiner (Milne, 2009), TAGME (Ferragina and Scaiella, 2010) and several annotation systems such as (Dill et al., 2003; Mihalcea and Csomai, 2007; Bunescu and Pasca, 2006) attach topics from Wikipedia (and other catalogs) to the documents by establishing the hard or soft links mentioned above.

Our goal is the following: Given a (ground set) collection V of topics organized in a pre-existing hierarchical DAG structure, and a collection D of documents, chose a size $K \in \mathbb{Z}_+$ representative subset of topics. Our approach is distinct from earlier work (e.g., (Kanungo et al., 2002; Blei et al., 2003)) where typically only a set of documents is classified and categorized in some way. We next provide a few definitions needed later in the paper.

Definition 1: Transitive Cover Γ): A topic t is said to cover a set of documents $\Gamma(t)$, called the transitive cover of the topic t, if for all documents $i \in \Gamma(t)$, either i is associated directly with topic t or with any of the descendant topics of t in the topic DAG. A natural extension of this definition to a set of topics T is defined as $\Gamma(T) = \cup_{t \in T} \Gamma(t)$.

[7] A simple greedy algorithm (Nemhauser et al., 1978) obtains a $1 - 1/e$ approximation guarantee for monotone submodular function maximization

Definition 2: Truncated Transitive Cover (Γ^α): This is a transitive cover of topic t, but with the limitation that the path length between a document and the topic t is not more than α. Hence, $|\Gamma^\alpha(t)| \leq |\Gamma(t)|$.

While our problem is closely related to clustering approaches, which consider the set of documents directly, there are some crucial differences. In particular, we focus on producing a clustering of documents where clusters are encouraged to honor a pre-defined DAG structured topic hierarchy. Existing agglomerative clustering algorithms focusing on the coverage of documents may not produce the desired clustering. To understand this, consider six documents d1, d2 ... d6 to be grouped into three clusters. There may be multiple ways to do this depending upon multiple aggregation paths present in the topic DAG: ((d1, d2), (d3, d4), (d5, d6)) or ((d1, d2, d3), (d4, d5), (d6)) or ((d1, d2, d3, d4), (d5), (d6)) or something else. Hence, we need more stringent measures to prefer one clustering over the others. Our work addresses this with a variety of quality criteria (coverage, diversity, specificity, clarity, relevance and fidelity, which are explained later in this paper) that are organically derived from well established submodular functions. And, most importantly, we learn the right mixture of these qualities to be enforced from the data itself. Furthermore, our approach also generalizes these clustering approaches, since one of the components in our mixture of submodular functions is defined via these unsupervised approaches, and maps a given clustering to a set of topics in the hierarchy.

3 Submodular Components and Learning

Summarization is the task of extracting information from a source that is both small in size but still representative. Our problem is different from traditional summarization tasks since we have an underlying DAG as a topic hierarchy that we wish to summarize in response to a subset of documents. Thus, a critical part of our problem is to take the graph structure into account while creating the summaries. Below, we identify properties we wish our summaries to posses.

Coverage: A summary set of topics should cover most of the documents. A document is said to be covered by a topic if there exists a path from the topic, going through intermediary descendant topics, to the document, i.e., the document is within the transitive cover of the topic.

Diversity: Summaries should be as diverse as possible, i.e., each summary topic should cover a unique set of documents. When a document is covered by more than one topic, that document is redundantly covered, e.g., "Finance" and "Banking" would be unlikely members of the same summary.

Summary qualities also involve "quality" notions, including:

Specificity/Clarity/Relevance/Coherence: These quality measures help us choose a set of topics that are neither too abstract nor overly specific. They ensure that the topics are clear and relevant to the documents that they represent. When additional information such as clustering (from LDA or other sources) and tagging (manual) documents is available, these quality criteria encourage the chosen topics to show resemblance (coherence) to those clustering/tagging in terms of transitive cover of documents they produce.

In the below, we define a variety of submodular functions that capture the above properties, and we then describe a large margin learning framework for learning convex mixtures of such components.

3.1 Submodular Components

3.1.1 Coverage Based Functions

Coverage components capture "coverage" of a set of documents.

Weighted Set Cover Function: Given a set of categories, $S \subseteq V$, define $\Gamma(S)$ as the set of documents covered — for each topic $s \in S$, $\Gamma(s) \subseteq D$ represents the documents covered by topic s and $\Gamma(S) = \cup_{s \in S} \Gamma(s)$. The weighted set cover function, defined as $f(S) = \sum_{d \in \Gamma(S)} w_d = w(\Gamma(S))$, assigns weights to the documents based on their relative importance (e.g., in Wikipedia disambiguation, the different documents could be ranked based on their priority).

Feature-based Functions: This class of function represents coverage in feature space. Given a set of categories $S \subseteq V$, and a set of features U, define $m_u(S)$ as the score associated with the set of categories S for feature $u \in U$. The feature set could represent, for example, the documents, in which case $m_u(S)$ represents the number of times document u is covered by the set S. U could also represent more complicated features. For example, in the context of Wikipedia disambiguation, U could represent TFIDF features over the documents. Feature based are then defined as $f(S) = \sum_{u \in U} \psi(m_u(S))$, where ψ is a concave (e.g., the square root) function. This function class has been successfully used in several applications (Kirchhoff and Bilmes, 2014; Wei et al., 2014a; Wei et al., 2014b).

3.1.2 Similarity based Functions

Similarity functions are defined through a similarity matrix $\mathbf{S} = \{s_{ij}\}_{i,j \in V}$. Given categories $i, j \in V$, similarity s_{ij} in our case can be defined as $s_{ij} = |\Gamma(i) \cap \Gamma(j)|$, i.e the number of documents commonly covered by both i and j.

Facility Location: The facility location function, defined as $f(S) = \sum_{i \in V} \max_{j \in S} s_{ij}$, is a natural model for k-medoids and exemplar based clustering, and has been used in several summarization problems (Tschiatschek et al., 2014; Wei et al., 2014a).

Penalty based diversity: A similarity matrix may be used to express a form of coverage of a set S but that is then penalized with a redundancy term, as in the following difference: $f(S) = \sum_{i \in V, j \in S} s_{ij} - \lambda \sum_{i \in S} \sum_{j \in S,} s_{i,j}$ (Lin and Bilmes, 2011)). Here $\lambda \in [0, 1]$. This function is submodular, but is not in general monotone, and has been used in document summarization (Lin and Bilmes, 2011), as a dispersion function (Borodin et al., 2012), and in image summarization (Tschiatschek et al., 2014).

3.1.3 Quality Control (QC) Functions

QC functions ensure a quality criteria is met by a set S of topics. We define the quality score of the set S as $F_q(S) = \sum_{s \in S} f_q(s)$, where $f_q(s)$ is the quality score of topic s for quality q. Therefore, $F_q(S)$ is a modular function in S. We investigate three types of quality control functions: Topic Specificity, Topic Clarity, and Topic Relevance.

Topic Specificity: The farther a topic is from the root of the DAG, the more specific it becomes. Topics higher up in the hierarchy are abstract and less specific. We therefore prefer topics low in the DAG, but lower topics also have less coverage. We define $f_{\text{specificity}}(s) = s_h$ where s_h is the height of topic s in the DAG. The root topic has height zero and the "height" increases as we move down the DAG in Figure 1.

Topic Clarity: Topic clarity is the fraction of descendant topics that cover one or more documents. If a topic has many descendant topics that do not cover any documents, it has less clarity. Formally, $f_{\text{clarity}}(s) = \frac{\sum_{t \in \text{descendants}(s)} [\![\Gamma(t) > 0]\!]}{|\text{descendants}(s)|}$, where $[\![.]\!]$ is the indicator function.

Topic Relevance: A topic is considered to be better related to a document if the number of hops needed to reach the document from that topic is lower. Given any set $A \subseteq D$ of document, and any topic $s \in V$, we can define $f_{\text{relevance}}(s|A) = \text{argmin}_\alpha \{\alpha : A \subseteq \Gamma^\alpha(s)\}$.

QC Functions As Barrier Modular Mixtures: We introduce a modular function for every QC function as follows

$$f^\alpha_{\text{specificity}}(s) = \begin{cases} 1 & \text{if the height of topic } s \text{ is at least } \alpha \\ 0 & \text{otherwise} \end{cases}$$

for every possible value of α. This creates a submodular mixture with as many components as the number of possible values of α. In our experiments with Wikipedia, we had α varying from 1 to 120 stepping by 1, adding 120 modular mixture components. Similarly, we define,

$$f^\beta_{\text{clarity}}(s) = \begin{cases} 1 & \text{if the clarity of topic } s \text{ is at least } \beta \\ 0 & \text{otherwise} \end{cases}$$

for every possible (discretized to make it countably finite) value of β. And,

$f^\gamma_{\text{relevance}}(s) = f_{\text{cov}}(s|\Gamma^\gamma(s))$, where $f_{\text{cov}}(.)$ is the coverage submodular function and $s|X$ indicates coverage of a topic s over a set of documents X. All these functions (modular and submodular terms) are added as mixture components in our learning framework to learn suitable weights for them. We then use these weights in our inference procedure to obtain a subset of topics as described in 3.2. We show from our experiments that this approach performs better than all other approaches and baselines.

3.1.4 Fidelity Functions

A function representing the fidelity of a set S to another reference set R is one that gets a large value when the set S represents the set R. Such a function scores inferred topics high when it resembles a reference set of topics and/or item clusters. The reference set in this case can be produced from other algorithms such as k-means, LDA and its variants or from a manually tagged corpus. Next we describe one such fidelity function.

Topic Coherence: This function scores a set of topics S high when the transitive cover (Definition 1) produced by the topics in S resembles the clusters of documents produced by an external source (k-means, LDA or manual). Given an external source that clusters the documents, producing T clusters $L_1, L_2, ..., L_T$ (for T topics), topic coherence is defined as: $f(S) = \sum_{t \in T} \max_{k \in S} w_{k,t}$ where $w_{k,t} = \text{harmonic\_mean}(w^p_{k,t}, w^r_{k,t})$ and $w^p_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|\Gamma(k)|}$ and $w^r_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|L_t|}$. Note that, $w^p_{k,t} \geq 0$ and $w^r_{k,t} \geq 0$ are the precision on recall of the resemblance and $w_{k,t}$ is the F1 measure. If the transitive cover of topics in S resembles the reference clusters L_t exactly, we attain maximum coherence (or fidelity). As the resemblance diminishes, the score decreases. The above function $f(S)$ is monotone submodular.

3.1.5 Mixture of Submodular Components:

Given the different classes of submodular functions above, we construct our submodular scoring functions $F_w(\cdot)$ as a convex combinations of these different submodular functions $f_1, f_2, \ldots, f_m$, above. In other words,

$$F_w(S) = \sum_{i=1}^{m} w_i f_i(S), \tag{2}$$

where $w = (w_1, \ldots, w_m)$, $w_i \geq 0, \sum_i w_i = 1$. The components f_i are submodular and assumed to be normalized: i.e., $f_i(\emptyset) = 0$, and $f_i(V) = 1$ for monotone functions and $\max_{A \subseteq V} f_i(A) \leq 1$ for non-monotone functions. A simple way to normalize a monotone submodular function is to define the component as $f_i(S)/f_i(V)$. This ensures that the components are *compatible* with each other. Obviously, the merit of the scoring function $F_w(\cdot)$ depends on the selection of the components.

3.2 Large Margin Learning

We optimize the weights w of the scoring function $F_w(\cdot)$ in a large-margin structured prediction framework. In this setting, we assume we have training data in the form of pairs of a set of documents, and a human generated summary as a set of topics. For example, in the case of Wikipedia disambiguation, we use the human generated disambiguation pages as the ground truth summary. We represent the set of ground-truth summaries as $\mathcal{S} = \{S_1, S_2, \cdots, S_N\}$. In large margin training, the weights are optimized such that ground-truth summaries $\mathcal{S}$ are separated from competitor summaries by a loss-dependent margin:

$$F_w(S) \geq F_w(S') + \mathcal{L}(S'), \quad \forall S \in \mathcal{S}, S' \in \mathcal{Y} \setminus \mathcal{S}, \tag{3}$$

where $\mathcal{L}(\cdot)$ is the loss function, and where $\mathcal{Y}$ is a structured output space (for example $\mathcal{Y}$ is the set of summaries that satisfy a certain budget B, i.e., $\mathcal{Y} = \{S' \subseteq V : |S'| \leq B\}$). We assume the loss to be normalized, $0 \leq \mathcal{L}(S') \leq 1, \forall S' \subseteq V$, to ensure that mixture and loss are calibrated. Equation (3) can be stated as $F_w(S) \geq \max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right], \forall S \in \mathcal{S}$ which is called *loss-augmented inference*. We introduce slack variables and minimize the regularized sum of slacks (Lin and Bilmes, 2012):

$$\min_{w \geq 0, \|w\|_1 = 1} \quad \sum_{S \in \mathcal{S}} \left[\max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right] - F_w(S)\right] + \frac{\lambda}{2}\|w\|_2^2, \tag{4}$$

where the non-negative orthant constraint, $w \geq 0$, ensures that the final mixture is submodular. Note a 2-norm regularizer is used on top of a 1-norm constraint $\|w\|_1 = 1$ which we interpret as a prior to encourage higher entropy, and thus more diverse mixture distributions. Tractability depends on the choice of the loss function. The parameters w are learnt using stochastic gradient descent as in (Tschiatschek et al., 2014).

3.3 Loss Functions

A natural choice of loss functions for our case can be derived from cluster evaluation metrics. Every inferred topic s induces a subset of documents, namely the transitive cover $\Gamma(s)$ of s. We compare these clusters with the clusters induced from the true topics in the training set and compute the loss.

In this paper, we use the Jaccard Index (JI) as a loss function. Let S be the inferred topics and T be the true topics. The Jaccard loss is defined as $\mathcal{L}_{\text{jaccard}}(S,T) = 1 - \frac{1}{k}\sum_{s \in S} \max_{t \in T} \frac{|\Gamma(s) \cap \Gamma(t)|}{|\Gamma(s) \cup \Gamma(t)|}$, where $k = |S| = |T|$ is the number of topics. When the clustering produced by the inferred and the true topics are similar, Jaccard loss is 0. When they are completely dissimilar, the loss is maximum, i.e., 1. Jaccard loss is a modular function.

3.4 Inference Algorithm: Greedy

Having learnt the weights for the mixture components, the resulting function $F_w(S) = \sum_{i=1}^{m} w_i f_i(S)$ is a submodular function. In the case when the individual components are themselves monotone (all our functions in fact are), $F_w(S)$ can be optimized by the accelerated greedy algorithm (Minoux, 1978). Thanks to submodularity, we can obtain near optimal solutions very efficiently. In case the functions are all monotone submodular, we can guarantee that the solution is within $1 - 1/e$ factor from the optimal solution.

4 Experimental Results

To validate our approach, we make use of Wikipedia category structure as a topic DAG and apply our technique to the task of automatic generation of Wikipedia disambiguation pages. We pre-processed the category graph to eliminate the cycles in order to make it a DAG. Each Wikipedia disambiguation page is manually created by Wikipedia editors by grouping a collection of Wikipedia articles into several groups. Each group is then assigned a name, which serves as a topic for the group. Typically, a disambiguation page segregates around 20-30 articles into 5-6 groups. Our goal is to measure how accurately we can recreate the groups for a disambiguation page and label them, given only the collection of articles mentioned in that disambiguation page (when actual groupings and labels are hidden).

4.1 Datasets

We parsed the contents of Wikipedia disambiguation pages and extracted disambiguation page names, article groups and group names. We collected about 8000 disambiguation pages that had at least four groups on them. Wikipedia category structure is used as the topic DAG. We eliminated few administrative categories such as "Hidden Categories", "Articles needing cleanup", and the like. The final DAG had about 1M topics and 3M links.

4.2 Evaluation Metrics

Every group of articles on the Wikipedia disambiguation page is assigned a name by the editors. Unfortunately, these names may not correspond to the Wikipedia category (topic) names. For example, one of the groups on the "Matrix" disambiguation page has a name "Business and government" and there is no Wikipedia category by that name. However, the group names generated by our (and baseline) method are from the Wikipedia categories (which forms our topic DAG). In addition, there can be multiple relevant names for a group. For example, a group on a disambiguation page may be called "Calculus", but an algorithm may rightly generate "Vector Calculus". Hence we cannot evaluate the accuracy of an algorithm just by matching the generated group names to those on the disambiguation page. To alleviate this problem, we adopt cluster-based evaluation metrics. We treat every group of articles generated by an algorithm under a topic for a disambiguation page as a cluster of articles. These are considered as *inferred* clusters for a disambiguation page. We compare them against the actual grouping of articles on the Wikipedia disambiguation page by treating those groups as *true* clusters. We can now adopt Jaccard Index, F1-measure, and NMI (Normalized Mutual Information) based cluster evaluation metrics described in (Manning et al., 2008). For each disambiguation page in the test set, we compute every metric score and then average it over all the disambiguation pages.

4.3 Methods Compared

We validated our approach by comparing against several baselines described below. We also compared two variations of our approach as described next. For each of these cases (baselines and two variations) we generated and compared the metrics (Jaccard Index, F1-measure and NMI) as described in the previous section.

KM_{docs}: K-Means algorithm run on articles as TF-IDF vectors of words. The number of clusters K is set to the number of true clusters on the Wikipedia disambiguation page.

$\text{KMed}_{\text{docs}}$: K-Medoids algorithm with articles as TF-IDF vectors of words. The number of clusters are set as in KM_{docs}.

$\text{KMed}_{\text{topics}}$: K-Medoids run on topics as TF-IDF vectors of words. The words for each topic is taken from the articles that are in the transitive cover of the topic.

LDA_{docs}: LDA algorithm with the number of topics set to the number of true clusters on the Wikipedia disambiguation page. Each article is then grouped under the highest probability topic.

SMML_{cov}: This is the submodular mixture learning case explained in section 3.1.5. Here we consider a mixture of all the submodular functions governing coverage, diversity, fidelity and QC functions. However, we exclude the similarity based functions described in section 3.1.2. Coverage based functions have a time complexity of $O(n)$ whereas similarity based functions are $O(n^2)$. By excluding similarity based functions, we can compare the quality of the results with and without $O(n^2)$ functions. We learn the mixture weights from the training set and use them during inference on the test set to subset K topics through the submodular maximization (Equation 1).

$\text{SMML}_{\text{cov+sim}}$: This case is similar to SMML_{cov} except that, we include similarity based submodular mixture components. This makes the inference time complexity $O(n^2)$.

We do not compare against HSLDA, PAM and few other techniques cited in the related work sections because they do not produce a subset of K summary topics — these are not directly comparable with our work.

4.4 Evaluation Results

We show that the submodular mixture learning and maximization approaches, i.e., SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other approaches in various metrics. In all these experiments, we performed 5 fold cross validation to learn the parameters from 80% of the disambiguation pages and evaluated on the rest of the 20%, in each fold.

In Figure 2a we summarize the results of the comparison of the methods mentioned above on Jaccard Index, F1 measure and NMI. Our proposed techniques SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other techniques consistently.

In Figures 2b and 2c we measure the number of test instances (i.e., disambiguation queries) in which each of the algorithms dominate (win) in evaluation metrics. In 60% of the disambiguation queries, SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ approaches

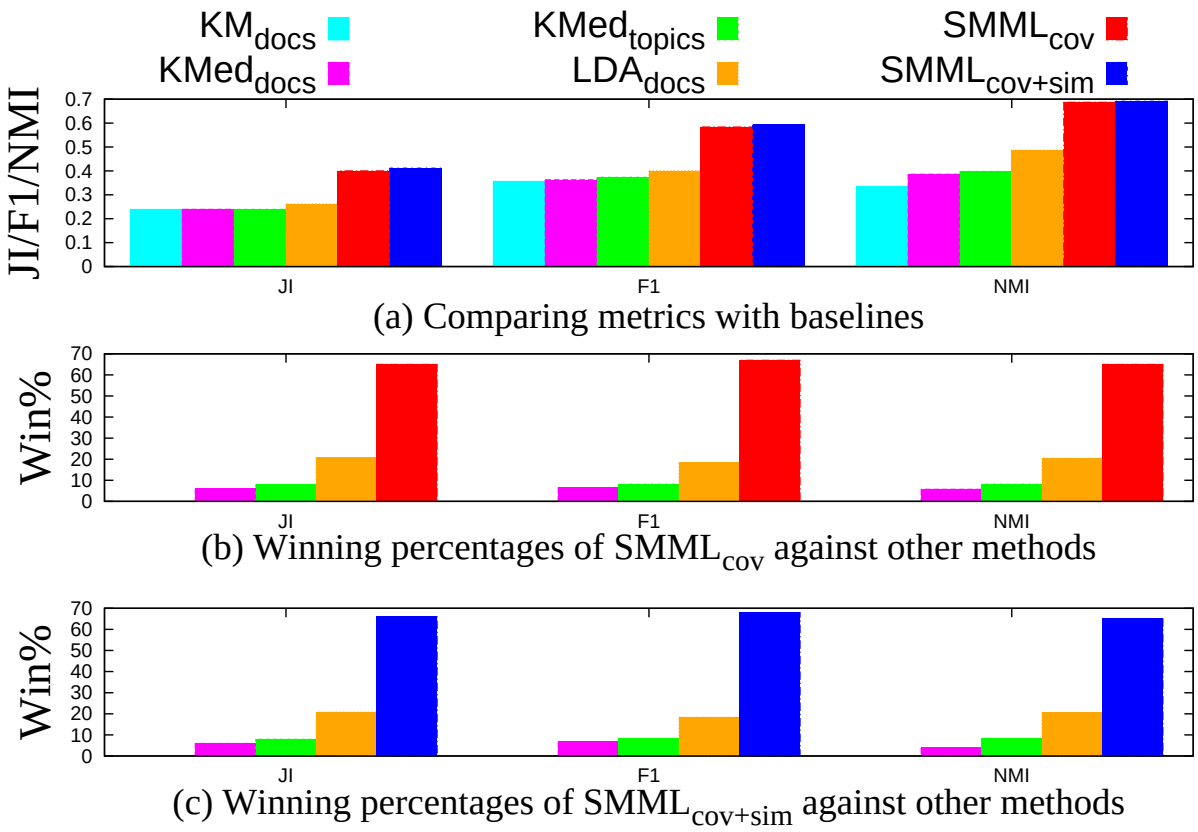

Figure 2: Comparison of techniques

produce higher JI, F1 and NMI than all other methods. This indicates that the clusters of articles produced by our technique resembles the clusters of articles present on the disambiguation page better than other techniques.

From Figures 2b and 2c it is clear that $O(n)$ time complexity based submodular mixture functions (SMML_{cov}) perform on par with $O(n^2)$ based functions ($\text{SMML}_{\text{cov+sim}}$), but at a greatly reduced execution time, demonstrating the sufficiency of $O(n)$ functions for our task. On the average, for each disambiguation query, SMML_{cov} took around 40 seconds (over 1M topics and 3M edges DAG) to infer the topics, whereas $\text{SMML}_{\text{cov+sim}}$ took around 35 minutes. Both these experiments were carried on a machine with 32 GB RAM, Eight-Core AMD Opteron(tm) Processor 2427.

5 Conclusions

We investigated a problem of summarizing topics over a massive topic DAG such that the summary set of topics produced represents the objects in the collection. This representation is characterized through various classes of submodular (and monotone) functions that captured coverage, similarity, diversity, specificity, clarity, relevance and fidelity of the topics. Currently we assume that the number of topics K is given as an input to our algorithm. It would be an interesting future problem to estimate the value of K automatically in our setting. As future work, we also plan to extend our techniques to produce a hierarchical summary of topics and scale it across heterogeneous collection of objects (from different domains) to bring all of them under the same topic DAG and investigate interesting cases thereon.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1162606, and by a Google, a Microsoft, and an Intel research award. Rishabh Iyer acknowledges support from the Microsoft Research Ph.D Fellowship.

References

Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. 2006. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, April.

W. Bi and J. T. Kwok. 2011. Multi-label classification on tree-and DAG-structured hierarchies. In *ICML*. ICML.

David M. Blei and John D. Lafferty. 2006. Correlated topic models. In *In Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. MIT Press.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, page 2003. MIT Press.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of Principles of Database Systems*, pages 155–166. ACM.

Razvan Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06), Trento, Italy*, pages 9–16, April.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. 2003. Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 178–186, New York, NY, USA. ACM.

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA.

R. Iyer, S. Jegelka, and J. Bilmes. 2013. Fast semidifferential-based submodular function optimization. ICML.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, Angela Y. Wu, Senior Member, and Senior Member. 2002. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:881–892.

D. Kempe, J. Kleinberg, and E. Tardos. 2003. Maximizing the spread of influence through a social network. In *SIGKDD*.

Katrin Kirchhoff and Jeff Bilmes. 2014. Submodularity for data selection in machine translation. October.

A. Krause and C. Guestrin. 2005. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of Uncertainity in Artificial Intelligence*. UAI.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 577–584, New York, NY, USA. ACM.

H. Lin and J. Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL*.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *The 49th Meeting of the Assoc. for Comp. Ling. Human Lang. Technologies (ACL/HLT-2011)*, Portland, OR, June.

H. Lin and J. Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, page 479490.

Hui Lin. 2012. *Submodularity in Natural Language Processing: Algorithms and Applications*. Ph.D. thesis, University of Washington, Dept. of EE.

Min ling Zhang and Zhi hua Zhou. 2007. Ml-knn: A lazy learning approach to multi-label learning. *PATTERN RECOGNITION*, 40:2007.

Arun S. Maiya, John P. Thompson, Francisco Loaiza-Lemos, and Robert M. Rolfe. 2013. Exploratory analysis of highly heterogeneous document collections. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1375–1383, New York, NY, USA. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Olena Medelyan, Ian H. Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the Wikipedia and AI workshop at AAAI-08*. AAAI.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 490–499, New York, NY, USA. ACM.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA. ACM.

David Milne. 2009. An open-source toolkit for mining wikipedia. In *In Proc. New Zealand Computer Science Research Student Conf*, page 2009.

Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, chapter 27, pages 234–243. Springer Berlin Heidelberg, Berlin/Heidelberg.

Mukund Narasimhan and Jeff Bilmes. 2004. PAC-learning bounded tree-width graphical models. In *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference (UAI-2004)*. Morgan Kaufmann Publishers, July.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294.

Adler J. Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. 2011. Hierarchically supervised latent dirichlet allocation. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *NIPS*, pages 2609–2617.

Juho Rousu, Craig Saunders, Sndor Szedmk, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626.

Jr. Silla, CarlosN. and AlexA. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Sebastian Tschiatschek, Rishabh Iyer, Hoachen Wei, and Jeff Bilmes. 2014. Learning Mixtures of Submodular Functions for Image Collection Summarization. In *Neural Information Processing Systems (NIPS)*.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer US.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2014a. Fast multi-stage submodular maximization. In *ICML*.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, Chris Bartels, and Jeff Bilmes. 2014b. Submodular subset selection for large-scale speech training data. *Proceedings of ICASSP, Florence, Italy*.

Torsten Zesch and Iryna Gurevych. 2007. Analysis of the wikipedia category graph for nlp applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, pages 1–8, Rochester, April. Association for Computational Linguistics.

Summarization of Multi-Document Topic Hierarchies using Submodular Mixtures

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, 40076, India
bairi@cse.iitb.ac.in

Rishabh Iyer
University of Washington
Seattle, WA-98175, USA
rkiyer@u.washington.edu

Ganesh Ramakrishnan
IIT Bombay
Mumbai, 40076, India
ganesh@cse.iitb.ac.in

Jeff Bilmes
University of Washington
Seattle, WA-98175, USA
bilmes@uw.edu

Abstract

We study the problem of summarizing DAG-structured topic hierarchies over a given set of documents. Example applications include automatically generating Wikipedia disambiguation pages for a set of articles, and generating candidate multi-labels for preparing machine learning datasets (e.g., for text classification, functional genomics, and image classification). Unlike previous work, which focuses on clustering the set of documents using the topic hierarchy as features, we directly pose the problem as a submodular optimization problem on a topic hierarchy using the documents as features. Desirable properties of the chosen topics include document coverage, specificity, topic diversity, and topic homogeneity, each of which, we show, is naturally modeled by a submodular function. Other information, provided say by unsupervised approaches such as LDA and its variants, can also be utilized by defining a submodular function that expresses coherence between the chosen topics and this information. We use a large-margin framework to learn convex mixtures over the set of submodular components. We empirically evaluate our method on the problem of automatically generating Wikipedia disambiguation pages using human generated clusterings as ground truth. We find that our framework improves upon several baselines according to a variety of standard evaluation metrics including the Jaccard Index, F1 score and NMI, and moreover, can be scaled to extremely large scale problems.

1 Introduction

Several real world machine learning applications involve hierarchy based categorization of topics for a set of objects. Objects could be, e.g., a set of documents for text classification, a set of genes in functional genomics, or a set of images in computer vision. One can often define a natural topic hierarchy to categorize these objects. For example, in text and image classification problems, each document or image is assigned a hierarchy of labels — a baseball page is assigned the labels "baseball" and "sports." Moreover, many of these applications, naturally have an existing topic hierarchy generated on the entire set of objects (Rousu et al., 2006; Barutcuoglu et al., 2006; ling Zhang and hua Zhou, 2007; Silla and Freitas, 2011; Tsoumakas et al., 2010).

Given a DAG-structured topic hierarchy and a subset of objects, we investigate the problem of finding a subset of DAG-structured topics that are induced by that subset (of objects). This problem arises naturally in several real world applications. For example, consider the problem of identifying appropriate label sets for a collection of articles. Several existing text collection datasets such as 20 Newsgroup[1], Reuters-21578[2] work with a predefined set of topics. We observe that these topic names are highly abstract[3] for the articles categorized under them. On the other hand, techniques proposed by systems such as Wikipedia Miner (Milne, 2009) and TAGME (Ferragina and Scaiella, 2010) generate several labels for each article in the dataset that are highly specific to the article. Collating all labels from all articles to create a label

[1] http://qwone.com/~jason/20Newsgroups/

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[3] Topic *Concept* is more abstract than the topic *Science* which is more abstract than the topic*Chemistry*

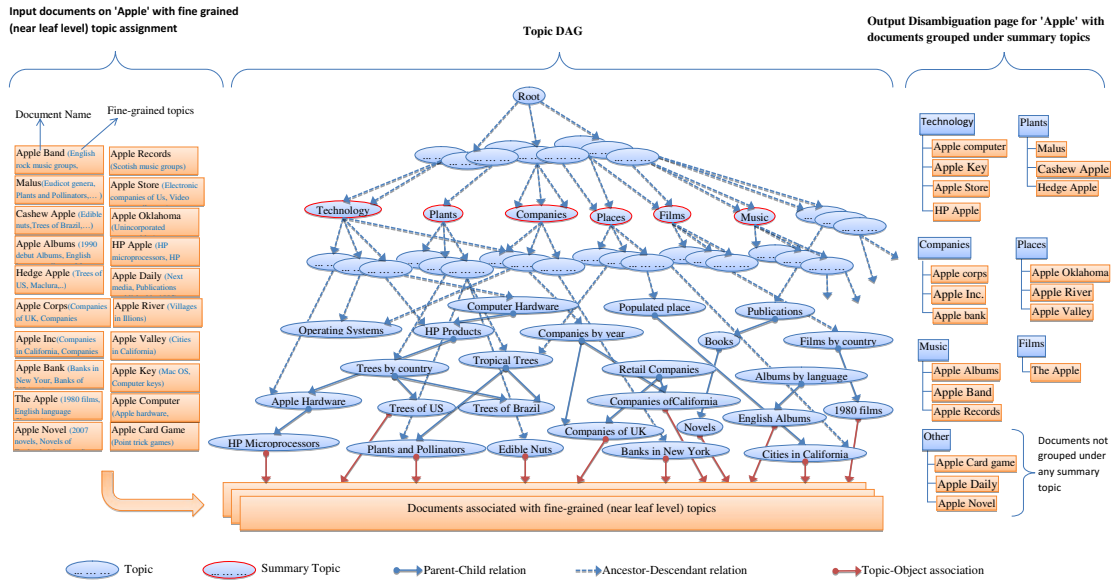

Figure 1: Topic Summarization overview. On the left, we show many documents related to Apple. In the middle, a Wikipedia category hierarchy shown as a topic DAG, links these documents at the leaf level. On the right, we show the output of our summarization process, which creates a set of summary topics (Plants, Technology, Companies, Films, Music and Places in this example) with the input documents classified under them.

set for the dataset can result in a large number of labels and become unmanageable. Our proposed techniques can summarize such large sets of labels into a smaller and more meaningful label sets using a DAG-structured topic hierarchy. This also holds for image classification problems and datasets like ImageNet (Deng et al., 2009). We use the term *summarize* to highlight the fact that the smaller label set semantically covers the larger label set. For example, the topics *Physics*, *Chemistry*, and *Mathematics* can be summarized into a topic *Science*.

A particularly important application of our work (and the one we use for our evaluations in Section 4) is the following: Given a collection of articles spanning different topics, but with similar titles, automatically generate a disambiguation page for those titles using the Wikipedia category hierarchy[4] as a topic DAG. Disambiguation pages[5] on Wikipedia are used to resolve conflicts in article titles that occur when a title is naturally associated with multiple articles on distinct topics. Each disambiguation page organizes articles into several groups, where the articles in each group pertain only to a specific topic. Disambiguations may be seen as paths in a hierarchy leading to different articles that arguably could have the same title. For example, the title *Apple*[6] can refer to a plant, a company, a film, a television show, a place, a technology, an album, a record label, and a newspaper daily. The problem then, is to organize the articles into multiple groups where each group contains articles of similar nature (topics) and has an appropriately discerned group heading. Figure 1 describes the topic summarization process for creation of the disambiguation page for "Apple".

All the above mentioned problems can be modeled as the problem of finding the most representative subset of topic nodes from a DAG-Structured topic hierarchy. We argue that many formulations of this problem are natural instances of submodular maximization, and provide a learning framework to create submodular mixtures to solve this problem. A set function $\mathrm{f}(.)$ is said to be submodular if for any element v and sets $A \subseteq B \subseteq V \setminus \{v\}$, where V represents the ground set of elements, $\mathrm{f}(A \cup \{v\}) - \mathrm{f}(A) \geq \mathrm{f}(B \cup \{v\}) - \mathrm{f}(B)$. This is called the diminishing returns property and states, informally, that adding an element to a smaller set increases the function value more than adding that element to a larger set. Submodular functions naturally model notions of coverage and diversity in applications, and therefore, a number of machine learning problems can be modeled as forms of submodular optimization (Kempe et al., 2003; Krause and Guestrin, 2005; Narasimhan and Bilmes, 2004; Iyer et al., 2013; Lin and Bilmes, 2012; Lin and Bilmes, 2010). In this paper, we investigate structured prediction methods for learn-

[4] http://en.wikipedia.org/wiki/Help:Categories

[5] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation

[6] http://en.wikipedia.org/wiki/Apple_(disambiguation)

ing weighted mixtures of submodular functions to summarize topics for a collection of objects using DAG-structured topic hierarchies. Throughout this paper we use the terms "topic" and "category" interchangeably.

1.1 Related Work

To the best of our knowledge, the specific problem we consider here is new. Previous work on identifying topics can be broadly categorized into one of the following types: a) cluster the objects and then identify names for the clusters; or b) dynamically identify topics (including hierarchical) for a set of objects. LDA (Blei et al., 2003) clusters the documents and simultaneously produces a set of topics into which the documents are clustered. In LDA, each document may be viewed as a mixture of various topics and the topic distribution is assumed to have a Dirichlet prior. LDA associates a group of high probability words to each identified topic. A name can be assigned to a topic by manually inspecting the words or using additional algorithms like (Mei et al., 2007; Maiya et al., 2013). LDA does not make use of existing topic hierarchies and correlation between topics. The Correlated Topic Model (Blei and Lafferty, 2006) induces a correlation structure between topics by using the logistic normal distribution instead of the Dirichlet. Another extension is the hierarchical LDA (Blei et al., 2004), where topics are joined together in a hierarchy by using the nested Chinese restaurant process. Nonparametric extensions of LDA include the Hierarchical Dirichlet Process (Teh et al., 2006) mixture model, which allows the number of topics to be unbounded and learnt from data and the Nested Chinese Restaurant Process which allows topics to be arranged in a hierarchy whose structure is learnt from data. In each of these approaches, unlike our proposed approach, an existing topic hierarchy is not used, nor is any additional object-topic information leveraged.

The pachinko allocation model (PAM)(Li and McCallum, 2006) captures arbitrary, nested, and possibly sparse correlations between topics using a DAG. The leaves of the DAG represent individual words in the vocabulary, while each interior node represents a correlation among its children, which may be words or other interior nodes (topics). PAM learns the probability distributions of words in a topic, subtopics in a topic, and topics in a document. We cannot, however, generate a subset of topics from a large existing topic DAG that can act as summary topics, using PAM.

HSLDA (Perotte et al., 2011) introduces a hierarchically supervised LDA model to infer hierarchical labels for a document. It assumes an existing label hierarchy in the form of a tree. The model infers one or more labels such that, if a label l is inferred as relevant to a document, then all the labels from l to the root of the tree are also inferred as relevant to the document. Our approach differs from HSLDA since: (1) we use the label hierarchy to infer a set of labels for a *group* of documents; (2) we do not enforce the label hierarchy to be a tree as it can be a DAG; and (3) generalizing HSLDA to use a DAG structured hierarchy and infer labels for a *group* of documents (e.g., combining into one big document) also may not help in solving our problem. HSLDA will apply *all* the relevant labels to the documents as per the classifier that it learns for every label. Moreover, the "root" label is always applied and it is very likely that many labels near the top level of the label hierarchy are also classified as relevant to the group of documents.

Wei and James (Bi and Kwok, 2011) present a hierarchical multi-label classification algorithm that can be used on both tree and DAG structured hierarchies. They formulate a search for the optimal consistent multi-label as the finding of the best subgraph in a tree/DAG. In our approach, we assume, individual documents are already associated with one or more topics and we find a consistent label set for a group of documents using the DAG structured topic hierarchy.

Medelyan et al. (Medelyan et al., 2008) and Ferragina et al. (Ferragina and Scaiella, 2010) detect topics for a document using Wikipedia article names and category names as the topic vocabulary. These systems are able to extract signals from a text document and identify Wikipedia articles and/or categories that optimally match the document and assign those article/category names as topics for the document. When run on a large collection of documents, these approaches generate enormous numbers of topics, a problem our proposed approach addresses.

1.2 Our Contributions

While most prior work discussed above focuses on the underlying set of documents, (*e.g.*, by clustering documents), we focus directly on the topics. In particular, we formulate the problem as subset selection on the set of topics within a DAG while simultaneously considering the documents to be categorized. Our method can scale to the colossal size of the DAG (1 million topics and 3 million correlation links between topics in Wikipedia). Moreover, our approach can

naturally incorporate outputs from many of the aforementioned algorithms. Our approach is based on submodular maximization and mixture learning, which has been successfully used in applications such as document summarization (Lin, 2012) and image summarization (Tschiatschek et al., 2014), but has never been applied to topic identification tasks or, more generally, DAG summarization.

We introduce a family of submodular functions to identify an appropriate set of topics from a DAG structured hierarchy of topics for a group of documents. We characterize this topic appropriateness through a set of desirable properties such as coverage, diversity, specificity, clarity, and relevance. Each of the submodular function components we consider are monotone, thereby ensuring a near optimal performance obtainable via a simple greedy algorithm for optimization.[7]. We also show how our technique naturally embodies outputs of other algorithms such as LDA, clustering, and classifications. Finally, we utilize a large margin formulation for learning mixtures of these submodular functions, and show how we can optimally learn them from training data.

Our approach demonstrates how to utilize the features collectively in the document space and the topic space to infer a set of topics. From an empirical perspective, we introduce and evaluate our approach on a dataset of around 8000 disambiguations that was extracted from Wikipedia and subsequently cleaned using the methods described in the experimentation section. We show that our learning framework outperforms many of the baselines, and is practical enough to be used on large corpora.

2 Problem Formulation

Let $G\,(V, E)$ be the DAG structured topic hierarchy with V topics. These topics are observed to have a parent child (*isa*) relationship forming a DAG. Let D be the set of documents that are associated with one or more of these topics. The middle portion of Figure 1 depicts a topic hierarchy with associated documents. The association links between the documents and topics can be hard or soft. In case of a hard link, a document is attached to a set of topics. Examples include multi-labeled documents. In case of a soft link, a document is associated with a topic with some degree of confidence (or probability). Furthermore, if a document is attached to a topic t, we assume that all the ancestor topics of t are also relevant for that document. This assumption has been employed in earlier works (Blei et al., 2004; Bi and Kwok, 2011; Rousu et al., 2006) as well. Given a budget of K, our objective is to choose a set of K topics from V, which best describe the documents in D. The notion of best describing topics is characterized through a set of desirable properties - coverage, diversity, specificity, clarity, relevance and fidelity - that K topics have to satisfy. The submodular functions that we introduce in the next section ensure these properties are satisfied. Formally, we solve the following discrete optimization problem:

$$S^* \in \underset{S \subseteq V : |S| \leq K}{\operatorname{argmax}} \sum_i w_i f_i(S) \qquad (1)$$

where, f_i are monotone submodular mixture components and $w_i \geq 0$ are the weights associated with those mixture components. Set S^* is the summary topics scored best.

It is easy to find massive (i.e., size in the order of million) DAG structured topic hierarchies in practice. Wikipedia's category hierarchy consists of more than 1M categories (topics) arranged hierarchically. In fact, they form a cyclic graph (Zesch and Gurevych, 2007). However, we can convert it to a DAG by eliminating the cycles as described in the supplementary material. YAGO (Suchanek et al., 2007) and Freebase (Bollacker et al., 2008) are other instances of massive topic hierarchies. The association of the documents with the existing topic hierarchy is also well studied. Systems such as WikipediaMiner (Milne, 2009), TAGME (Ferragina and Scaiella, 2010) and several annotation systems such as (Dill et al., 2003; Mihalcea and Csomai, 2007; Bunescu and Pasca, 2006) attach topics from Wikipedia (and other catalogs) to the documents by establishing the hard or soft links mentioned above.

Our goal is the following: Given a (ground set) collection V of topics organized in a pre-existing hierarchical DAG structure, and a collection D of documents, chose a size $K \in \mathbb{Z}_+$ representative subset of topics. Our approach is distinct from earlier work (e.g., (Kanungo et al., 2002; Blei et al., 2003)) where typically only a set of documents is classified and categorized in some way. We next provide a few definitions needed later in the paper.

Definition 1: Transitive Cover Γ): A topic t is said to cover a set of documents $\Gamma(t)$, called the transitive cover of the topic t, if for all documents $i \in \Gamma(t)$, either i is associated directly with topic t or with any of the descendant topics of t in the topic DAG. A natural extension of this definition to a set of topics T is defined as $\Gamma(T) = \cup_{t \in T} \Gamma(t)$.

[7] A simple greedy algorithm (Nemhauser et al., 1978) obtains a $1 - 1/e$ approximation guarantee for monotone submodular function maximization

Definition 2: Truncated Transitive Cover (Γ^α): This is a transitive cover of topic t, but with the limitation that the path length between a document and the topic t is not more than α. Hence, $|\Gamma^\alpha(t)| \leq |\Gamma(t)|$.

While our problem is closely related to clustering approaches, which consider the set of documents directly, there are some crucial differences. In particular, we focus on producing a clustering of documents where clusters are encouraged to honor a pre-defined DAG structured topic hierarchy. Existing agglomerative clustering algorithms focusing on the coverage of documents may not produce the desired clustering. To understand this, consider six documents d1, d2 ... d6 to be grouped into three clusters. There may be multiple ways to do this depending upon multiple aggregation paths present in the topic DAG: ((d1, d2), (d3, d4), (d5, d6)) or ((d1, d2, d3), (d4, d5), (d6)) or ((d1, d2, d3, d4), (d5), (d6)) or something else. Hence, we need more stringent measures to prefer one clustering over the others. Our work addresses this with a variety of quality criteria (coverage, diversity, specificity, clarity, relevance and fidelity, which are explained later in this paper) that are organically derived from well established submodular functions. And, most importantly, we learn the right mixture of these qualities to be enforced from the data itself. Furthermore, our approach also generalizes these clustering approaches, since one of the components in our mixture of submodular functions is defined via these unsupervised approaches, and maps a given clustering to a set of topics in the hierarchy.

3 Submodular Components and Learning

Summarization is the task of extracting information from a source that is both small in size but still representative. Our problem is different from traditional summarization tasks since we have an underlying DAG as a topic hierarchy that we wish to summarize in response to a subset of documents. Thus, a critical part of our problem is to take the graph structure into account while creating the summaries. Below, we identify properties we wish our summaries to posses.

Coverage: A summary set of topics should cover most of the documents. A document is said to be covered by a topic if there exists a path from the topic, going through intermediary descendant topics, to the document, i.e., the document is within the transitive cover of the topic.

Diversity: Summaries should be as diverse as possible, i.e., each summary topic should cover a unique set of documents. When a document is covered by more than one topic, that document is redundantly covered, e.g., "Finance" and "Banking" would be unlikely members of the same summary.

Summary qualities also involve "quality" notions, including:

Specificity/Clarity/Relevance/Coherence: These quality measures help us choose a set of topics that are neither too abstract nor overly specific. They ensure that the topics are clear and relevant to the documents that they represent. When additional information such as clustering (from LDA or other sources) and tagging (manual) documents is available, these quality criteria encourage the chosen topics to show resemblance (coherence) to those clustering/tagging in terms of transitive cover of documents they produce.

In the below, we define a variety of submodular functions that capture the above properties, and we then describe a large margin learning framework for learning convex mixtures of such components.

3.1 Submodular Components

3.1.1 Coverage Based Functions

Coverage components capture "coverage" of a set of documents.

Weighted Set Cover Function: Given a set of categories, $S \subseteq V$, define $\Gamma(S)$ as the set of documents covered — for each topic $s \in S$, $\Gamma(s) \subseteq D$ represents the documents covered by topic s and $\Gamma(S) = \cup_{s \in S} \Gamma(s)$. The weighted set cover function, defined as $f(S) = \sum_{d \in \Gamma(S)} w_d = w(\Gamma(S))$, assigns weights to the documents based on their relative importance (e.g., in Wikipedia disambiguation, the different documents could be ranked based on their priority).

Feature-based Functions: This class of function represents coverage in feature space. Given a set of categories $S \subseteq V$, and a set of features U, define $m_u(S)$ as the score associated with the set of categories S for feature $u \in U$. The feature set could represent, for example, the documents, in which case $m_u(S)$ represents the number of times document u is covered by the set S. U could also represent more complicated features. For example, in the context of Wikipedia disambiguation, U could represent TFIDF features over the documents. Feature based are then defined as $f(S) = \sum_{u \in U} \psi(m_u(S))$, where ψ is a concave (e.g., the square root) function. This function class has been successfully used in several applications (Kirchhoff and Bilmes, 2014; Wei et al., 2014a; Wei et al., 2014b).

3.1.2 Similarity based Functions

Similarity functions are defined through a similarity matrix $\mathbf{S} = \{s_{ij}\}_{i,j \in V}$. Given categories $i, j \in V$, similarity s_{ij} in our case can be defined as $s_{ij} = |\Gamma(i) \cap \Gamma(j)|$, i.e the number of documents commonly covered by both i and j.

Facility Location: The facility location function, defined as $f(S) = \sum_{i \in V} \max_{j \in S} s_{ij}$, is a natural model for k-medoids and exemplar based clustering, and has been used in several summarization problems (Tschiatschek et al., 2014; Wei et al., 2014a).

Penalty based diversity: A similarity matrix may be used to express a form of coverage of a set S but that is then penalized with a redundancy term, as in the following difference: $f(S) = \sum_{i \in V, j \in S} s_{ij} - \lambda \sum_{i \in S} \sum_{j \in S,} s_{i,j}$ (Lin and Bilmes, 2011)). Here $\lambda \in [0, 1]$. This function is submodular, but is not in general monotone, and has been used in document summarization (Lin and Bilmes, 2011), as a dispersion function (Borodin et al., 2012), and in image summarization (Tschiatschek et al., 2014).

3.1.3 Quality Control (QC) Functions

QC functions ensure a quality criteria is met by a set S of topics. We define the quality score of the set S as $F_q(S) = \sum_{s \in S} f_q(s)$, where $f_q(s)$ is the quality score of topic s for quality q. Therefore, $F_q(S)$ is a modular function in S. We investigate three types of quality control functions: Topic Specificity, Topic Clarity, and Topic Relevance.

Topic Specificity: The farther a topic is from the root of the DAG, the more specific it becomes. Topics higher up in the hierarchy are abstract and less specific. We therefore prefer topics low in the DAG, but lower topics also have less coverage. We define $f_{\text{specificity}}(s) = s_h$ where s_h is the height of topic s in the DAG. The root topic has height zero and the "height" increases as we move down the DAG in Figure 1.

Topic Clarity: Topic clarity is the fraction of descendant topics that cover one or more documents. If a topic has many descendant topics that do not cover any documents, it has less clarity. Formally, $f_{\text{clarity}}(s) = \frac{\sum_{t \in \text{descendants}(s)} [\![\Gamma(t) > 0]\!]}{|\text{descendants}(s)|}$, where $[\![.]\!]$ is the indicator function.

Topic Relevance: A topic is considered to be better related to a document if the number of hops needed to reach the document from that topic is lower. Given any set $A \subseteq D$ of document, and any topic $s \in V$, we can define $f_{\text{relevance}}(s|A) = \text{argmin}_\alpha \{\alpha : A \subseteq \Gamma^\alpha(s)\}$.

QC Functions As Barrier Modular Mixtures: We introduce a modular function for every QC function as follows

$$f^\alpha_{\text{specificity}}(s) = \begin{cases} 1 & \text{if the height of topic } s \text{ is at least } \alpha \\ 0 & \text{otherwise} \end{cases}$$

for every possible value of α. This creates a submodular mixture with as many components as the number of possible values of α. In our experiments with Wikipedia, we had α varying from 1 to 120 stepping by 1, adding 120 modular mixture components. Similarly, we define,

$$f^\beta_{\text{clarity}}(s) = \begin{cases} 1 & \text{if the clarity of topic } s \text{ is at least } \beta \\ 0 & \text{otherwise} \end{cases}$$

for every possible (discretized to make it countably finite) value of β. And,

$f^\gamma_{\text{relevance}}(s) = f_{\text{cov}}(s|\Gamma^\gamma(s))$, where $f_{\text{cov}}(.)$ is the coverage submodular function and $s|X$ indicates coverage of a topic s over a set of documents X. All these functions (modular and submodular terms) are added as mixture components in our learning framework to learn suitable weights for them. We then use these weights in our inference procedure to obtain a subset of topics as described in 3.2. We show from our experiments that this approach performs better than all other approaches and baselines.

3.1.4 Fidelity Functions

A function representing the fidelity of a set S to another reference set R is one that gets a large value when the set S represents the set R. Such a function scores inferred topics high when it resembles a reference set of topics and/or item clusters. The reference set in this case can be produced from other algorithms such as k-means, LDA and its variants or from a manually tagged corpus. Next we describe one such fidelity function.

Topic Coherence: This function scores a set of topics S high when the transitive cover (Definition 1) produced by the topics in S resembles the clusters of documents produced by an external source (k-means, LDA or manual). Given an external source that clusters the documents, producing T clusters $L_1, L_2, ..., L_T$ (for T topics), topic coherence is defined as: $f(S) = \sum_{t \in T} \max_{k \in S} w_{k,t}$ where $w_{k,t} = \text{harmonic\_mean}(w^p_{k,t}, w^r_{k,t})$ and $w^p_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|\Gamma(k)|}$ and $w^r_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|L_t|}$. Note that, $w^p_{k,t} \geq 0$ and $w^r_{k,t} \geq 0$ are the precision on recall of the resemblance and $w_{k,t}$ is the F1 measure. If the transitive cover of topics in S resembles the reference clusters L_t exactly, we attain maximum coherence (or fidelity). As the resemblance diminishes, the score decreases. The above function $f(S)$ is monotone submodular.

3.1.5 Mixture of Submodular Components:

Given the different classes of submodular functions above, we construct our submodular scoring functions $F_w(\cdot)$ as a convex combinations of these different submodular functions $f_1, f_2, \ldots, f_m$, above. In other words,

$$F_w(S) = \sum_{i=1}^{m} w_i f_i(S), \tag{2}$$

where $w = (w_1, \ldots, w_m)$, $w_i \geq 0, \sum_i w_i = 1$. The components f_i are submodular and assumed to be normalized: i.e., $f_i(\emptyset) = 0$, and $f_i(V) = 1$ for monotone functions and $\max_{A \subseteq V} f_i(A) \leq 1$ for non-monotone functions. A simple way to normalize a monotone submodular function is to define the component as $f_i(S)/f_i(V)$. This ensures that the components are *compatible* with each other. Obviously, the merit of the scoring function $F_w(\cdot)$ depends on the selection of the components.

3.2 Large Margin Learning

We optimize the weights w of the scoring function $F_w(\cdot)$ in a large-margin structured prediction framework. In this setting, we assume we have training data in the form of pairs of a set of documents, and a human generated summary as a set of topics. For example, in the case of Wikipedia disambiguation, we use the human generated disambiguation pages as the ground truth summary. We represent the set of ground-truth summaries as $\mathcal{S} = \{S_1, S_2, \cdots, S_N\}$. In large margin training, the weights are optimized such that ground-truth summaries $\mathcal{S}$ are separated from competitor summaries by a loss-dependent margin:

$$F_w(S) \geq F_w(S') + \mathcal{L}(S'), \quad \forall S \in \mathcal{S}, S' \in \mathcal{Y} \setminus \mathcal{S}, \tag{3}$$

where $\mathcal{L}(\cdot)$ is the loss function, and where $\mathcal{Y}$ is a structured output space (for example $\mathcal{Y}$ is the set of summaries that satisfy a certain budget B, i.e., $\mathcal{Y} = \{S' \subseteq V : |S'| \leq B\}$). We assume the loss to be normalized, $0 \leq \mathcal{L}(S') \leq 1, \forall S' \subseteq V$, to ensure that mixture and loss are calibrated. Equation (3) can be stated as $F_w(S) \geq \max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right], \forall S \in \mathcal{S}$ which is called *loss-augmented inference*. We introduce slack variables and minimize the regularized sum of slacks (Lin and Bilmes, 2012):

$$\min_{w \geq 0, \|w\|_1 = 1} \quad \sum_{S \in \mathcal{S}} \left[\max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right] - F_w(S)\right] + \frac{\lambda}{2}\|w\|_2^2, \tag{4}$$

where the non-negative orthant constraint, $w \geq 0$, ensures that the final mixture is submodular. Note a 2-norm regularizer is used on top of a 1-norm constraint $\|w\|_1 = 1$ which we interpret as a prior to encourage higher entropy, and thus more diverse mixture distributions. Tractability depends on the choice of the loss function. The parameters w are learnt using stochastic gradient descent as in (Tschiatschek et al., 2014).

3.3 Loss Functions

A natural choice of loss functions for our case can be derived from cluster evaluation metrics. Every inferred topic s induces a subset of documents, namely the transitive cover $\Gamma(s)$ of s. We compare these clusters with the clusters induced from the true topics in the training set and compute the loss.

In this paper, we use the Jaccard Index (JI) as a loss function. Let S be the inferred topics and T be the true topics. The Jaccard loss is defined as $\mathcal{L}_{\text{jaccard}}(S,T) = 1 - \frac{1}{k}\sum_{s \in S} \max_{t \in T} \frac{|\Gamma(s) \cap \Gamma(t)|}{|\Gamma(s) \cup \Gamma(t)|}$, where $k = |S| = |T|$ is the number of topics. When the clustering produced by the inferred and the true topics are similar, Jaccard loss is 0. When they are completely dissimilar, the loss is maximum, i.e., 1. Jaccard loss is a modular function.

3.4 Inference Algorithm: Greedy

Having learnt the weights for the mixture components, the resulting function $F_w(S) = \sum_{i=1}^{m} w_i f_i(S)$ is a submodular function. In the case when the individual components are themselves monotone (all our functions in fact are), $F_w(S)$ can be optimized by the accelerated greedy algorithm (Minoux, 1978). Thanks to submodularity, we can obtain near optimal solutions very efficiently. In case the functions are all monotone submodular, we can guarantee that the solution is within $1 - 1/e$ factor from the optimal solution.

4 Experimental Results

To validate our approach, we make use of Wikipedia category structure as a topic DAG and apply our technique to the task of automatic generation of Wikipedia disambiguation pages. We pre-processed the category graph to eliminate the cycles in order to make it a DAG. Each Wikipedia disambiguation page is manually created by Wikipedia editors by grouping a collection of Wikipedia articles into several groups. Each group is then assigned a name, which serves as a topic for the group. Typically, a disambiguation page segregates around 20-30 articles into 5-6 groups. Our goal is to measure how accurately we can recreate the groups for a disambiguation page and label them, given only the collection of articles mentioned in that disambiguation page (when actual groupings and labels are hidden).

4.1 Datasets

We parsed the contents of Wikipedia disambiguation pages and extracted disambiguation page names, article groups and group names. We collected about 8000 disambiguation pages that had at least four groups on them. Wikipedia category structure is used as the topic DAG. We eliminated few administrative categories such as "Hidden Categories", "Articles needing cleanup", and the like. The final DAG had about 1M topics and 3M links.

4.2 Evaluation Metrics

Every group of articles on the Wikipedia disambiguation page is assigned a name by the editors. Unfortunately, these names may not correspond to the Wikipedia category (topic) names. For example, one of the groups on the "Matrix" disambiguation page has a name "Business and government" and there is no Wikipedia category by that name. However, the group names generated by our (and baseline) method are from the Wikipedia categories (which forms our topic DAG). In addition, there can be multiple relevant names for a group. For example, a group on a disambiguation page may be called "Calculus", but an algorithm may rightly generate "Vector Calculus". Hence we cannot evaluate the accuracy of an algorithm just by matching the generated group names to those on the disambiguation page. To alleviate this problem, we adopt cluster-based evaluation metrics. We treat every group of articles generated by an algorithm under a topic for a disambiguation page as a cluster of articles. These are considered as *inferred* clusters for a disambiguation page. We compare them against the actual grouping of articles on the Wikipedia disambiguation page by treating those groups as *true* clusters. We can now adopt Jaccard Index, F1-measure, and NMI (Normalized Mutual Information) based cluster evaluation metrics described in (Manning et al., 2008). For each disambiguation page in the test set, we compute every metric score and then average it over all the disambiguation pages.

4.3 Methods Compared

We validated our approach by comparing against several baselines described below. We also compared two variations of our approach as described next. For each of these cases (baselines and two variations) we generated and compared the metrics (Jaccard Index, F1-measure and NMI) as described in the previous section.

KM_{docs}: K-Means algorithm run on articles as TF-IDF vectors of words. The number of clusters K is set to the number of true clusters on the Wikipedia disambiguation page.

$\text{KMed}_{\text{docs}}$: K-Medoids algorithm with articles as TF-IDF vectors of words. The number of clusters are set as in KM_{docs}.

$\text{KMed}_{\text{topics}}$: K-Medoids run on topics as TF-IDF vectors of words. The words for each topic is taken from the articles that are in the transitive cover of the topic.

LDA_{docs}: LDA algorithm with the number of topics set to the number of true clusters on the Wikipedia disambiguation page. Each article is then grouped under the highest probability topic.

SMML_{cov}: This is the submodular mixture learning case explained in section 3.1.5. Here we consider a mixture of all the submodular functions governing coverage, diversity, fidelity and QC functions. However, we exclude the similarity based functions described in section 3.1.2. Coverage based functions have a time complexity of $O(n)$ whereas similarity based functions are $O(n^2)$. By excluding similarity based functions, we can compare the quality of the results with and without $O(n^2)$ functions. We learn the mixture weights from the training set and use them during inference on the test set to subset K topics through the submodular maximization (Equation 1).

$\text{SMML}_{\text{cov+sim}}$: This case is similar to SMML_{cov} except that, we include similarity based submodular mixture components. This makes the inference time complexity $O(n^2)$.

We do not compare against HSLDA, PAM and few other techniques cited in the related work sections because they do not produce a subset of K summary topics — these are not directly comparable with our work.

4.4 Evaluation Results

We show that the submodular mixture learning and maximization approaches, i.e., SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other approaches in various metrics. In all these experiments, we performed 5 fold cross validation to learn the parameters from 80% of the disambiguation pages and evaluated on the rest of the 20%, in each fold.

In Figure 2a we summarize the results of the comparison of the methods mentioned above on Jaccard Index, F1 measure and NMI. Our proposed techniques SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other techniques consistently.

In Figures 2b and 2c we measure the number of test instances (i.e., disambiguation queries) in which each of the algorithms dominate (win) in evaluation metrics. In 60% of the disambiguation queries, SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ approaches

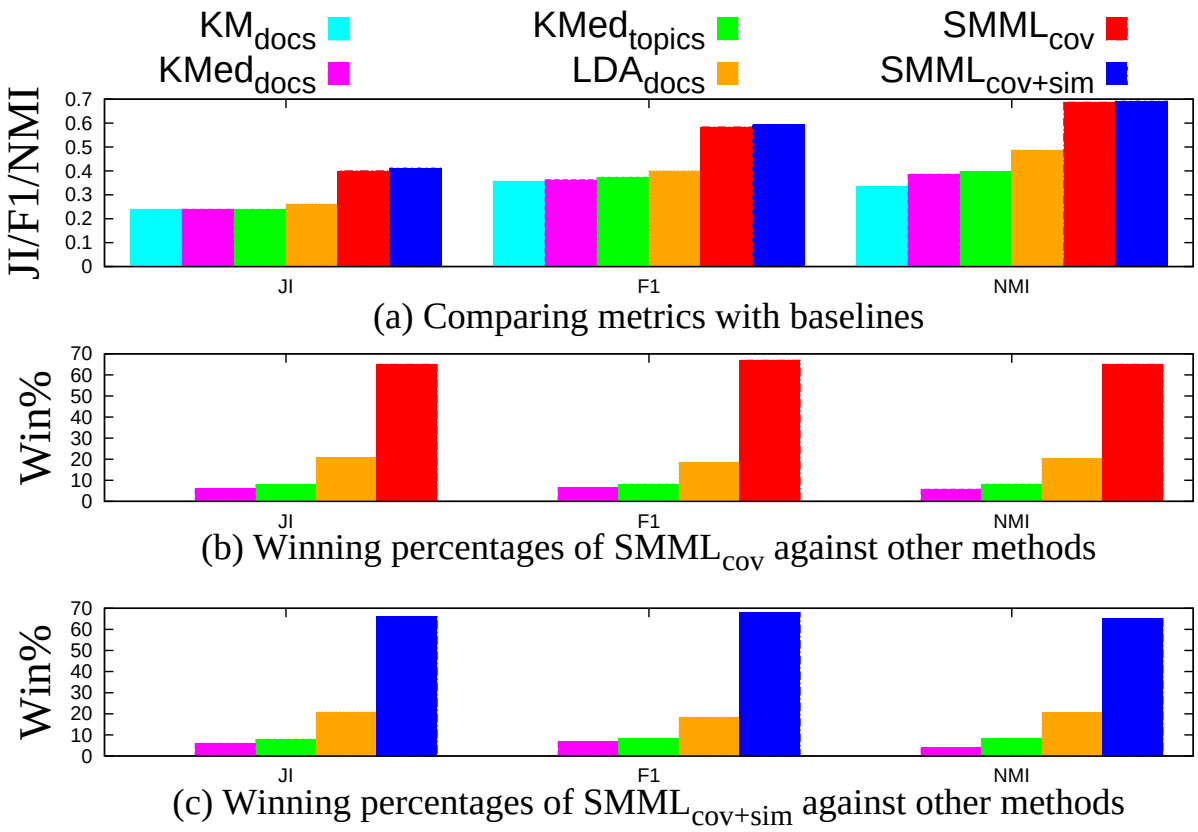

Figure 2: Comparison of techniques

produce higher JI, F1 and NMI than all other methods. This indicates that the clusters of articles produced by our technique resembles the clusters of articles present on the disambiguation page better than other techniques.

From Figures 2b and 2c it is clear that $O(n)$ time complexity based submodular mixture functions (SMML_{cov}) perform on par with $O(n^2)$ based functions ($\text{SMML}_{\text{cov+sim}}$), but at a greatly reduced execution time, demonstrating the sufficiency of $O(n)$ functions for our task. On the average, for each disambiguation query, SMML_{cov} took around 40 seconds (over 1M topics and 3M edges DAG) to infer the topics, whereas $\text{SMML}_{\text{cov+sim}}$ took around 35 minutes. Both these experiments were carried on a machine with 32 GB RAM, Eight-Core AMD Opteron(tm) Processor 2427.

5 Conclusions

We investigated a problem of summarizing topics over a massive topic DAG such that the summary set of topics produced represents the objects in the collection. This representation is characterized through various classes of submodular (and monotone) functions that captured coverage, similarity, diversity, specificity, clarity, relevance and fidelity of the topics. Currently we assume that the number of topics K is given as an input to our algorithm. It would be an interesting future problem to estimate the value of K automatically in our setting. As future work, we also plan to extend our techniques to produce a hierarchical summary of topics and scale it across heterogeneous collection of objects (from different domains) to bring all of them under the same topic DAG and investigate interesting cases thereon.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1162606, and by a Google, a Microsoft, and an Intel research award. Rishabh Iyer acknowledges support from the Microsoft Research Ph.D Fellowship.

References

Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. 2006. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, April.

W. Bi and J. T. Kwok. 2011. Multi-label classification on tree-and DAG-structured hierarchies. In *ICML*. ICML.

David M. Blei and John D. Lafferty. 2006. Correlated topic models. In *In Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. MIT Press.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, page 2003. MIT Press.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of Principles of Database Systems*, pages 155–166. ACM.

Razvan Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06), Trento, Italy*, pages 9–16, April.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. 2003. Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 178–186, New York, NY, USA. ACM.

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA.

R. Iyer, S. Jegelka, and J. Bilmes. 2013. Fast semidifferential-based submodular function optimization. ICML.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, Angela Y. Wu, Senior Member, and Senior Member. 2002. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:881–892.

D. Kempe, J. Kleinberg, and E. Tardos. 2003. Maximizing the spread of influence through a social network. In *SIGKDD*.

Katrin Kirchhoff and Jeff Bilmes. 2014. Submodularity for data selection in machine translation. October.

A. Krause and C. Guestrin. 2005. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of Uncertainity in Artificial Intelligence*. UAI.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 577–584, New York, NY, USA. ACM.

H. Lin and J. Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL*.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *The 49th Meeting of the Assoc. for Comp. Ling. Human Lang. Technologies (ACL/HLT-2011)*, Portland, OR, June.

H. Lin and J. Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, page 479490.

Hui Lin. 2012. *Submodularity in Natural Language Processing: Algorithms and Applications*. Ph.D. thesis, University of Washington, Dept. of EE.

Min ling Zhang and Zhi hua Zhou. 2007. Ml-knn: A lazy learning approach to multi-label learning. *PATTERN RECOGNITION*, 40:2007.

Arun S. Maiya, John P. Thompson, Francisco Loaiza-Lemos, and Robert M. Rolfe. 2013. Exploratory analysis of highly heterogeneous document collections. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1375–1383, New York, NY, USA. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Olena Medelyan, Ian H. Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the Wikipedia and AI workshop at AAAI-08*. AAAI.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 490–499, New York, NY, USA. ACM.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA. ACM.

David Milne. 2009. An open-source toolkit for mining wikipedia. In *In Proc. New Zealand Computer Science Research Student Conf*, page 2009.

Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, chapter 27, pages 234–243. Springer Berlin Heidelberg, Berlin/Heidelberg.

Mukund Narasimhan and Jeff Bilmes. 2004. PAC-learning bounded tree-width graphical models. In *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference (UAI-2004)*. Morgan Kaufmann Publishers, July.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294.

Adler J. Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. 2011. Hierarchically supervised latent dirichlet allocation. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *NIPS*, pages 2609–2617.

Juho Rousu, Craig Saunders, Sndor Szedmk, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626.

Jr. Silla, CarlosN. and AlexA. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Sebastian Tschiatschek, Rishabh Iyer, Hoachen Wei, and Jeff Bilmes. 2014. Learning Mixtures of Submodular Functions for Image Collection Summarization. In *Neural Information Processing Systems (NIPS)*.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer US.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2014a. Fast multi-stage submodular maximization. In *ICML*.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, Chris Bartels, and Jeff Bilmes. 2014b. Submodular subset selection for large-scale speech training data. *Proceedings of ICASSP, Florence, Italy*.

Torsten Zesch and Iryna Gurevych. 2007. Analysis of the wikipedia category graph for nlp applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, pages 1–8, Rochester, April. Association for Computational Linguistics.

Summarization of Multi-Document Topic Hierarchies using Submodular Mixtures

Ramakrishna B Bairi
IITB-Monash Research Academy
IIT Bombay
Mumbai, 40076, India
bairi@cse.iitb.ac.in

Rishabh Iyer
University of Washington
Seattle, WA-98175, USA
rkiyer@u.washington.edu

Ganesh Ramakrishnan
IIT Bombay
Mumbai, 40076, India
ganesh@cse.iitb.ac.in

Jeff Bilmes
University of Washington
Seattle, WA-98175, USA
bilmes@uw.edu

Abstract

We study the problem of summarizing DAG-structured topic hierarchies over a given set of documents. Example applications include automatically generating Wikipedia disambiguation pages for a set of articles, and generating candidate multi-labels for preparing machine learning datasets (e.g., for text classification, functional genomics, and image classification). Unlike previous work, which focuses on clustering the set of documents using the topic hierarchy as features, we directly pose the problem as a submodular optimization problem on a topic hierarchy using the documents as features. Desirable properties of the chosen topics include document coverage, specificity, topic diversity, and topic homogeneity, each of which, we show, is naturally modeled by a submodular function. Other information, provided say by unsupervised approaches such as LDA and its variants, can also be utilized by defining a submodular function that expresses coherence between the chosen topics and this information. We use a large-margin framework to learn convex mixtures over the set of submodular components. We empirically evaluate our method on the problem of automatically generating Wikipedia disambiguation pages using human generated clusterings as ground truth. We find that our framework improves upon several baselines according to a variety of standard evaluation metrics including the Jaccard Index, F1 score and NMI, and moreover, can be scaled to extremely large scale problems.

1 Introduction

Several real world machine learning applications involve hierarchy based categorization of topics for a set of objects. Objects could be, e.g., a set of documents for text classification, a set of genes in functional genomics, or a set of images in computer vision. One can often define a natural topic hierarchy to categorize these objects. For example, in text and image classification problems, each document or image is assigned a hierarchy of labels — a baseball page is assigned the labels "baseball" and "sports." Moreover, many of these applications, naturally have an existing topic hierarchy generated on the entire set of objects (Rousu et al., 2006; Barutcuoglu et al., 2006; ling Zhang and hua Zhou, 2007; Silla and Freitas, 2011; Tsoumakas et al., 2010).

Given a DAG-structured topic hierarchy and a subset of objects, we investigate the problem of finding a subset of DAG-structured topics that are induced by that subset (of objects). This problem arises naturally in several real world applications. For example, consider the problem of identifying appropriate label sets for a collection of articles. Several existing text collection datasets such as 20 Newsgroup[1], Reuters-21578[2] work with a predefined set of topics. We observe that these topic names are highly abstract[3] for the articles categorized under them. On the other hand, techniques proposed by systems such as Wikipedia Miner (Milne, 2009) and TAGME (Ferragina and Scaiella, 2010) generate several labels for each article in the dataset that are highly specific to the article. Collating all labels from all articles to create a label

[1] http://qwone.com/~jason/20Newsgroups/

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/

[3] Topic *Concept* is more abstract than the topic *Science* which is more abstract than the topic*Chemistry*

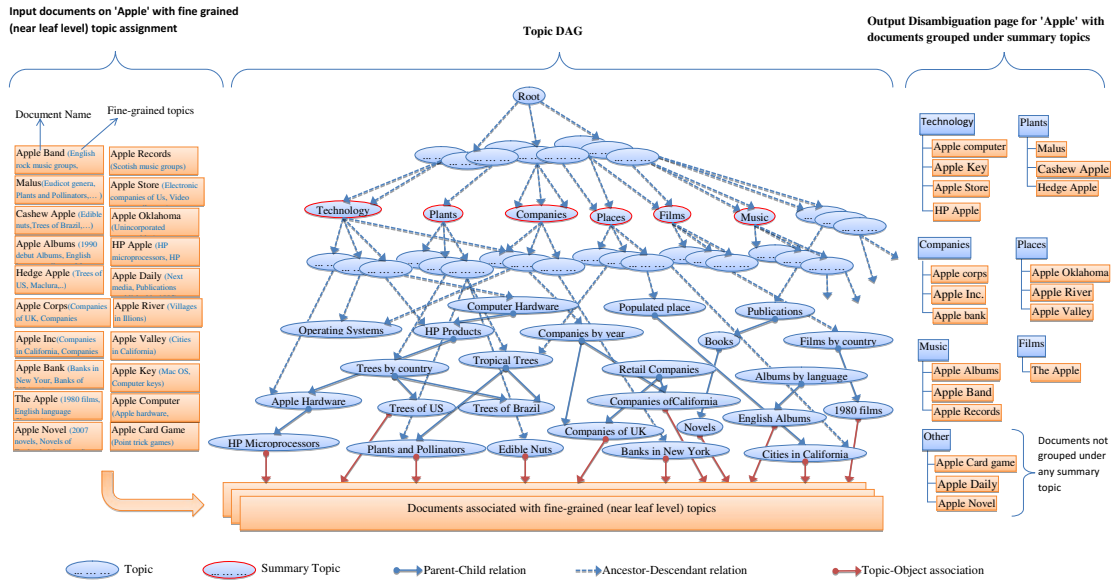

Figure 1: Topic Summarization overview. On the left, we show many documents related to Apple. In the middle, a Wikipedia category hierarchy shown as a topic DAG, links these documents at the leaf level. On the right, we show the output of our summarization process, which creates a set of summary topics (Plants, Technology, Companies, Films, Music and Places in this example) with the input documents classified under them.

set for the dataset can result in a large number of labels and become unmanageable. Our proposed techniques can summarize such large sets of labels into a smaller and more meaningful label sets using a DAG-structured topic hierarchy. This also holds for image classification problems and datasets like ImageNet (Deng et al., 2009). We use the term *summarize* to highlight the fact that the smaller label set semantically covers the larger label set. For example, the topics *Physics*, *Chemistry*, and *Mathematics* can be summarized into a topic *Science*.

A particularly important application of our work (and the one we use for our evaluations in Section 4) is the following: Given a collection of articles spanning different topics, but with similar titles, automatically generate a disambiguation page for those titles using the Wikipedia category hierarchy[4] as a topic DAG. Disambiguation pages[5] on Wikipedia are used to resolve conflicts in article titles that occur when a title is naturally associated with multiple articles on distinct topics. Each disambiguation page organizes articles into several groups, where the articles in each group pertain only to a specific topic. Disambiguations may be seen as paths in a hierarchy leading to different articles that arguably could have the same title. For example, the title *Apple*[6] can refer to a plant, a company, a film, a television show, a place, a technology, an album, a record label, and a newspaper daily. The problem then, is to organize the articles into multiple groups where each group contains articles of similar nature (topics) and has an appropriately discerned group heading. Figure 1 describes the topic summarization process for creation of the disambiguation page for "Apple".

All the above mentioned problems can be modeled as the problem of finding the most representative subset of topic nodes from a DAG-Structured topic hierarchy. We argue that many formulations of this problem are natural instances of submodular maximization, and provide a learning framework to create submodular mixtures to solve this problem. A set function $\mathrm{f}(.)$ is said to be submodular if for any element v and sets $A \subseteq B \subseteq V \setminus \{v\}$, where V represents the ground set of elements, $\mathrm{f}(A \cup \{v\}) - \mathrm{f}(A) \geq \mathrm{f}(B \cup \{v\}) - \mathrm{f}(B)$. This is called the diminishing returns property and states, informally, that adding an element to a smaller set increases the function value more than adding that element to a larger set. Submodular functions naturally model notions of coverage and diversity in applications, and therefore, a number of machine learning problems can be modeled as forms of submodular optimization (Kempe et al., 2003; Krause and Guestrin, 2005; Narasimhan and Bilmes, 2004; Iyer et al., 2013; Lin and Bilmes, 2012; Lin and Bilmes, 2010). In this paper, we investigate structured prediction methods for learn-

[4] http://en.wikipedia.org/wiki/Help:Categories

[5] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation

[6] http://en.wikipedia.org/wiki/Apple_(disambiguation)

ing weighted mixtures of submodular functions to summarize topics for a collection of objects using DAG-structured topic hierarchies. Throughout this paper we use the terms "topic" and "category" interchangeably.

1.1 Related Work

To the best of our knowledge, the specific problem we consider here is new. Previous work on identifying topics can be broadly categorized into one of the following types: a) cluster the objects and then identify names for the clusters; or b) dynamically identify topics (including hierarchical) for a set of objects. LDA (Blei et al., 2003) clusters the documents and simultaneously produces a set of topics into which the documents are clustered. In LDA, each document may be viewed as a mixture of various topics and the topic distribution is assumed to have a Dirichlet prior. LDA associates a group of high probability words to each identified topic. A name can be assigned to a topic by manually inspecting the words or using additional algorithms like (Mei et al., 2007; Maiya et al., 2013). LDA does not make use of existing topic hierarchies and correlation between topics. The Correlated Topic Model (Blei and Lafferty, 2006) induces a correlation structure between topics by using the logistic normal distribution instead of the Dirichlet. Another extension is the hierarchical LDA (Blei et al., 2004), where topics are joined together in a hierarchy by using the nested Chinese restaurant process. Nonparametric extensions of LDA include the Hierarchical Dirichlet Process (Teh et al., 2006) mixture model, which allows the number of topics to be unbounded and learnt from data and the Nested Chinese Restaurant Process which allows topics to be arranged in a hierarchy whose structure is learnt from data. In each of these approaches, unlike our proposed approach, an existing topic hierarchy is not used, nor is any additional object-topic information leveraged.

The pachinko allocation model (PAM)(Li and McCallum, 2006) captures arbitrary, nested, and possibly sparse correlations between topics using a DAG. The leaves of the DAG represent individual words in the vocabulary, while each interior node represents a correlation among its children, which may be words or other interior nodes (topics). PAM learns the probability distributions of words in a topic, subtopics in a topic, and topics in a document. We cannot, however, generate a subset of topics from a large existing topic DAG that can act as summary topics, using PAM.

HSLDA (Perotte et al., 2011) introduces a hierarchically supervised LDA model to infer hierarchical labels for a document. It assumes an existing label hierarchy in the form of a tree. The model infers one or more labels such that, if a label l is inferred as relevant to a document, then all the labels from l to the root of the tree are also inferred as relevant to the document. Our approach differs from HSLDA since: (1) we use the label hierarchy to infer a set of labels for a *group* of documents; (2) we do not enforce the label hierarchy to be a tree as it can be a DAG; and (3) generalizing HSLDA to use a DAG structured hierarchy and infer labels for a *group* of documents (e.g., combining into one big document) also may not help in solving our problem. HSLDA will apply *all* the relevant labels to the documents as per the classifier that it learns for every label. Moreover, the "root" label is always applied and it is very likely that many labels near the top level of the label hierarchy are also classified as relevant to the group of documents.

Wei and James (Bi and Kwok, 2011) present a hierarchical multi-label classification algorithm that can be used on both tree and DAG structured hierarchies. They formulate a search for the optimal consistent multi-label as the finding of the best subgraph in a tree/DAG. In our approach, we assume, individual documents are already associated with one or more topics and we find a consistent label set for a group of documents using the DAG structured topic hierarchy.

Medelyan et al. (Medelyan et al., 2008) and Ferragina et al. (Ferragina and Scaiella, 2010) detect topics for a document using Wikipedia article names and category names as the topic vocabulary. These systems are able to extract signals from a text document and identify Wikipedia articles and/or categories that optimally match the document and assign those article/category names as topics for the document. When run on a large collection of documents, these approaches generate enormous numbers of topics, a problem our proposed approach addresses.

1.2 Our Contributions

While most prior work discussed above focuses on the underlying set of documents, (*e.g.*, by clustering documents), we focus directly on the topics. In particular, we formulate the problem as subset selection on the set of topics within a DAG while simultaneously considering the documents to be categorized. Our method can scale to the colossal size of the DAG (1 million topics and 3 million correlation links between topics in Wikipedia). Moreover, our approach can

naturally incorporate outputs from many of the aforementioned algorithms. Our approach is based on submodular maximization and mixture learning, which has been successfully used in applications such as document summarization (Lin, 2012) and image summarization (Tschiatschek et al., 2014), but has never been applied to topic identification tasks or, more generally, DAG summarization.

We introduce a family of submodular functions to identify an appropriate set of topics from a DAG structured hierarchy of topics for a group of documents. We characterize this topic appropriateness through a set of desirable properties such as coverage, diversity, specificity, clarity, and relevance. Each of the submodular function components we consider are monotone, thereby ensuring a near optimal performance obtainable via a simple greedy algorithm for optimization.[7]. We also show how our technique naturally embodies outputs of other algorithms such as LDA, clustering, and classifications. Finally, we utilize a large margin formulation for learning mixtures of these submodular functions, and show how we can optimally learn them from training data.

Our approach demonstrates how to utilize the features collectively in the document space and the topic space to infer a set of topics. From an empirical perspective, we introduce and evaluate our approach on a dataset of around 8000 disambiguations that was extracted from Wikipedia and subsequently cleaned using the methods described in the experimentation section. We show that our learning framework outperforms many of the baselines, and is practical enough to be used on large corpora.

2 Problem Formulation

Let $G\,(V, E)$ be the DAG structured topic hierarchy with V topics. These topics are observed to have a parent child (*isa*) relationship forming a DAG. Let D be the set of documents that are associated with one or more of these topics. The middle portion of Figure 1 depicts a topic hierarchy with associated documents. The association links between the documents and topics can be hard or soft. In case of a hard link, a document is attached to a set of topics. Examples include multi-labeled documents. In case of a soft link, a document is associated with a topic with some degree of confidence (or probability). Furthermore, if a document is attached to a topic t, we assume that all the ancestor topics of t are also relevant for that document. This assumption has been employed in earlier works (Blei et al., 2004; Bi and Kwok, 2011; Rousu et al., 2006) as well. Given a budget of K, our objective is to choose a set of K topics from V, which best describe the documents in D. The notion of best describing topics is characterized through a set of desirable properties - coverage, diversity, specificity, clarity, relevance and fidelity - that K topics have to satisfy. The submodular functions that we introduce in the next section ensure these properties are satisfied. Formally, we solve the following discrete optimization problem:

$$S^* \in \underset{S \subseteq V : |S| \leq K}{\operatorname{argmax}} \sum_i w_i f_i(S) \qquad (1)$$

where, f_i are monotone submodular mixture components and $w_i \geq 0$ are the weights associated with those mixture components. Set S^* is the summary topics scored best.

It is easy to find massive (i.e., size in the order of million) DAG structured topic hierarchies in practice. Wikipedia's category hierarchy consists of more than 1M categories (topics) arranged hierarchically. In fact, they form a cyclic graph (Zesch and Gurevych, 2007). However, we can convert it to a DAG by eliminating the cycles as described in the supplementary material. YAGO (Suchanek et al., 2007) and Freebase (Bollacker et al., 2008) are other instances of massive topic hierarchies. The association of the documents with the existing topic hierarchy is also well studied. Systems such as WikipediaMiner (Milne, 2009), TAGME (Ferragina and Scaiella, 2010) and several annotation systems such as (Dill et al., 2003; Mihalcea and Csomai, 2007; Bunescu and Pasca, 2006) attach topics from Wikipedia (and other catalogs) to the documents by establishing the hard or soft links mentioned above.

Our goal is the following: Given a (ground set) collection V of topics organized in a pre-existing hierarchical DAG structure, and a collection D of documents, chose a size $K \in \mathbb{Z}_+$ representative subset of topics. Our approach is distinct from earlier work (e.g., (Kanungo et al., 2002; Blei et al., 2003)) where typically only a set of documents is classified and categorized in some way. We next provide a few definitions needed later in the paper.

Definition 1: Transitive Cover Γ): A topic t is said to cover a set of documents $\Gamma(t)$, called the transitive cover of the topic t, if for all documents $i \in \Gamma(t)$, either i is associated directly with topic t or with any of the descendant topics of t in the topic DAG. A natural extension of this definition to a set of topics T is defined as $\Gamma(T) = \cup_{t \in T} \Gamma(t)$.

[7] A simple greedy algorithm (Nemhauser et al., 1978) obtains a $1 - 1/e$ approximation guarantee for monotone submodular function maximization

Definition 2: Truncated Transitive Cover (Γ^α): This is a transitive cover of topic t, but with the limitation that the path length between a document and the topic t is not more than α. Hence, $|\Gamma^\alpha(t)| \leq |\Gamma(t)|$.

While our problem is closely related to clustering approaches, which consider the set of documents directly, there are some crucial differences. In particular, we focus on producing a clustering of documents where clusters are encouraged to honor a pre-defined DAG structured topic hierarchy. Existing agglomerative clustering algorithms focusing on the coverage of documents may not produce the desired clustering. To understand this, consider six documents d1, d2 ... d6 to be grouped into three clusters. There may be multiple ways to do this depending upon multiple aggregation paths present in the topic DAG: ((d1, d2), (d3, d4), (d5, d6)) or ((d1, d2, d3), (d4, d5), (d6)) or ((d1, d2, d3, d4), (d5), (d6)) or something else. Hence, we need more stringent measures to prefer one clustering over the others. Our work addresses this with a variety of quality criteria (coverage, diversity, specificity, clarity, relevance and fidelity, which are explained later in this paper) that are organically derived from well established submodular functions. And, most importantly, we learn the right mixture of these qualities to be enforced from the data itself. Furthermore, our approach also generalizes these clustering approaches, since one of the components in our mixture of submodular functions is defined via these unsupervised approaches, and maps a given clustering to a set of topics in the hierarchy.

3 Submodular Components and Learning

Summarization is the task of extracting information from a source that is both small in size but still representative. Our problem is different from traditional summarization tasks since we have an underlying DAG as a topic hierarchy that we wish to summarize in response to a subset of documents. Thus, a critical part of our problem is to take the graph structure into account while creating the summaries. Below, we identify properties we wish our summaries to posses.

Coverage: A summary set of topics should cover most of the documents. A document is said to be covered by a topic if there exists a path from the topic, going through intermediary descendant topics, to the document, i.e., the document is within the transitive cover of the topic.

Diversity: Summaries should be as diverse as possible, i.e., each summary topic should cover a unique set of documents. When a document is covered by more than one topic, that document is redundantly covered, e.g., "Finance" and "Banking" would be unlikely members of the same summary.

Summary qualities also involve "quality" notions, including:

Specificity/Clarity/Relevance/Coherence: These quality measures help us choose a set of topics that are neither too abstract nor overly specific. They ensure that the topics are clear and relevant to the documents that they represent. When additional information such as clustering (from LDA or other sources) and tagging (manual) documents is available, these quality criteria encourage the chosen topics to show resemblance (coherence) to those clustering/tagging in terms of transitive cover of documents they produce.

In the below, we define a variety of submodular functions that capture the above properties, and we then describe a large margin learning framework for learning convex mixtures of such components.

3.1 Submodular Components

3.1.1 Coverage Based Functions

Coverage components capture "coverage" of a set of documents.

Weighted Set Cover Function: Given a set of categories, $S \subseteq V$, define $\Gamma(S)$ as the set of documents covered — for each topic $s \in S$, $\Gamma(s) \subseteq D$ represents the documents covered by topic s and $\Gamma(S) = \cup_{s \in S} \Gamma(s)$. The weighted set cover function, defined as $f(S) = \sum_{d \in \Gamma(S)} w_d = w(\Gamma(S))$, assigns weights to the documents based on their relative importance (e.g., in Wikipedia disambiguation, the different documents could be ranked based on their priority).

Feature-based Functions: This class of function represents coverage in feature space. Given a set of categories $S \subseteq V$, and a set of features U, define $m_u(S)$ as the score associated with the set of categories S for feature $u \in U$. The feature set could represent, for example, the documents, in which case $m_u(S)$ represents the number of times document u is covered by the set S. U could also represent more complicated features. For example, in the context of Wikipedia disambiguation, U could represent TFIDF features over the documents. Feature based are then defined as $f(S) = \sum_{u \in U} \psi(m_u(S))$, where ψ is a concave (e.g., the square root) function. This function class has been successfully used in several applications (Kirchhoff and Bilmes, 2014; Wei et al., 2014a; Wei et al., 2014b).

3.1.2 Similarity based Functions

Similarity functions are defined through a similarity matrix $\mathbf{S} = \{s_{ij}\}_{i,j \in V}$. Given categories $i, j \in V$, similarity s_{ij} in our case can be defined as $s_{ij} = |\Gamma(i) \cap \Gamma(j)|$, i.e the number of documents commonly covered by both i and j.

Facility Location: The facility location function, defined as $f(S) = \sum_{i \in V} \max_{j \in S} s_{ij}$, is a natural model for k-medoids and exemplar based clustering, and has been used in several summarization problems (Tschiatschek et al., 2014; Wei et al., 2014a).

Penalty based diversity: A similarity matrix may be used to express a form of coverage of a set S but that is then penalized with a redundancy term, as in the following difference: $f(S) = \sum_{i \in V, j \in S} s_{ij} - \lambda \sum_{i \in S} \sum_{j \in S,} s_{i,j}$ (Lin and Bilmes, 2011)). Here $\lambda \in [0, 1]$. This function is submodular, but is not in general monotone, and has been used in document summarization (Lin and Bilmes, 2011), as a dispersion function (Borodin et al., 2012), and in image summarization (Tschiatschek et al., 2014).

3.1.3 Quality Control (QC) Functions

QC functions ensure a quality criteria is met by a set S of topics. We define the quality score of the set S as $F_q(S) = \sum_{s \in S} f_q(s)$, where $f_q(s)$ is the quality score of topic s for quality q. Therefore, $F_q(S)$ is a modular function in S. We investigate three types of quality control functions: Topic Specificity, Topic Clarity, and Topic Relevance.

Topic Specificity: The farther a topic is from the root of the DAG, the more specific it becomes. Topics higher up in the hierarchy are abstract and less specific. We therefore prefer topics low in the DAG, but lower topics also have less coverage. We define $f_{\text{specificity}}(s) = s_h$ where s_h is the height of topic s in the DAG. The root topic has height zero and the "height" increases as we move down the DAG in Figure 1.

Topic Clarity: Topic clarity is the fraction of descendant topics that cover one or more documents. If a topic has many descendant topics that do not cover any documents, it has less clarity. Formally, $f_{\text{clarity}}(s) = \frac{\sum_{t \in \text{descendants}(s)} [\![\Gamma(t) > 0]\!]}{|\text{descendants}(s)|}$, where $[\![.]\!]$ is the indicator function.

Topic Relevance: A topic is considered to be better related to a document if the number of hops needed to reach the document from that topic is lower. Given any set $A \subseteq D$ of document, and any topic $s \in V$, we can define $f_{\text{relevance}}(s|A) = \text{argmin}_\alpha \{\alpha : A \subseteq \Gamma^\alpha(s)\}$.

QC Functions As Barrier Modular Mixtures: We introduce a modular function for every QC function as follows

$$f^\alpha_{\text{specificity}}(s) = \begin{cases} 1 & \text{if the height of topic } s \text{ is at least } \alpha \\ 0 & \text{otherwise} \end{cases}$$

for every possible value of α. This creates a submodular mixture with as many components as the number of possible values of α. In our experiments with Wikipedia, we had α varying from 1 to 120 stepping by 1, adding 120 modular mixture components. Similarly, we define,

$$f^\beta_{\text{clarity}}(s) = \begin{cases} 1 & \text{if the clarity of topic } s \text{ is at least } \beta \\ 0 & \text{otherwise} \end{cases}$$

for every possible (discretized to make it countably finite) value of β. And,

$f^\gamma_{\text{relevance}}(s) = f_{\text{cov}}(s|\Gamma^\gamma(s))$, where $f_{\text{cov}}(.)$ is the coverage submodular function and $s|X$ indicates coverage of a topic s over a set of documents X. All these functions (modular and submodular terms) are added as mixture components in our learning framework to learn suitable weights for them. We then use these weights in our inference procedure to obtain a subset of topics as described in 3.2. We show from our experiments that this approach performs better than all other approaches and baselines.

3.1.4 Fidelity Functions

A function representing the fidelity of a set S to another reference set R is one that gets a large value when the set S represents the set R. Such a function scores inferred topics high when it resembles a reference set of topics and/or item clusters. The reference set in this case can be produced from other algorithms such as k-means, LDA and its variants or from a manually tagged corpus. Next we describe one such fidelity function.

Topic Coherence: This function scores a set of topics S high when the transitive cover (Definition 1) produced by the topics in S resembles the clusters of documents produced by an external source (k-means, LDA or manual). Given an external source that clusters the documents, producing T clusters $L_1, L_2, ..., L_T$ (for T topics), topic coherence is defined as: $f(S) = \sum_{t \in T} \max_{k \in S} w_{k,t}$ where $w_{k,t} = \text{harmonic\_mean}(w^p_{k,t}, w^r_{k,t})$ and $w^p_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|\Gamma(k)|}$ and $w^r_{k,t} = \frac{|\Gamma(k) \cap L_t|}{|L_t|}$. Note that, $w^p_{k,t} \geq 0$ and $w^r_{k,t} \geq 0$ are the precision on recall of the resemblance and $w_{k,t}$ is the F1 measure. If the transitive cover of topics in S resembles the reference clusters L_t exactly, we attain maximum coherence (or fidelity). As the resemblance diminishes, the score decreases. The above function $f(S)$ is monotone submodular.

3.1.5 Mixture of Submodular Components:

Given the different classes of submodular functions above, we construct our submodular scoring functions $F_w(\cdot)$ as a convex combinations of these different submodular functions $f_1, f_2, \ldots, f_m$, above. In other words,

$$F_w(S) = \sum_{i=1}^{m} w_i f_i(S), \tag{2}$$

where $w = (w_1, \ldots, w_m)$, $w_i \geq 0, \sum_i w_i = 1$. The components f_i are submodular and assumed to be normalized: i.e., $f_i(\emptyset) = 0$, and $f_i(V) = 1$ for monotone functions and $\max_{A \subseteq V} f_i(A) \leq 1$ for non-monotone functions. A simple way to normalize a monotone submodular function is to define the component as $f_i(S)/f_i(V)$. This ensures that the components are *compatible* with each other. Obviously, the merit of the scoring function $F_w(\cdot)$ depends on the selection of the components.

3.2 Large Margin Learning

We optimize the weights w of the scoring function $F_w(\cdot)$ in a large-margin structured prediction framework. In this setting, we assume we have training data in the form of pairs of a set of documents, and a human generated summary as a set of topics. For example, in the case of Wikipedia disambiguation, we use the human generated disambiguation pages as the ground truth summary. We represent the set of ground-truth summaries as $\mathcal{S} = \{S_1, S_2, \cdots, S_N\}$. In large margin training, the weights are optimized such that ground-truth summaries $\mathcal{S}$ are separated from competitor summaries by a loss-dependent margin:

$$F_w(S) \geq F_w(S') + \mathcal{L}(S'), \quad \forall S \in \mathcal{S}, S' \in \mathcal{Y} \setminus \mathcal{S}, \tag{3}$$

where $\mathcal{L}(\cdot)$ is the loss function, and where $\mathcal{Y}$ is a structured output space (for example $\mathcal{Y}$ is the set of summaries that satisfy a certain budget B, i.e., $\mathcal{Y} = \{S' \subseteq V : |S'| \leq B\}$). We assume the loss to be normalized, $0 \leq \mathcal{L}(S') \leq 1, \forall S' \subseteq V$, to ensure that mixture and loss are calibrated. Equation (3) can be stated as $F_w(S) \geq \max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right], \forall S \in \mathcal{S}$ which is called *loss-augmented inference*. We introduce slack variables and minimize the regularized sum of slacks (Lin and Bilmes, 2012):

$$\min_{w \geq 0, \|w\|_1 = 1} \quad \sum_{S \in \mathcal{S}} \left[\max_{S' \in \mathcal{Y}} \left[F_w(S') + \mathcal{L}(S')\right] - F_w(S)\right] + \frac{\lambda}{2}\|w\|_2^2, \tag{4}$$

where the non-negative orthant constraint, $w \geq 0$, ensures that the final mixture is submodular. Note a 2-norm regularizer is used on top of a 1-norm constraint $\|w\|_1 = 1$ which we interpret as a prior to encourage higher entropy, and thus more diverse mixture distributions. Tractability depends on the choice of the loss function. The parameters w are learnt using stochastic gradient descent as in (Tschiatschek et al., 2014).

3.3 Loss Functions

A natural choice of loss functions for our case can be derived from cluster evaluation metrics. Every inferred topic s induces a subset of documents, namely the transitive cover $\Gamma(s)$ of s. We compare these clusters with the clusters induced from the true topics in the training set and compute the loss.

In this paper, we use the Jaccard Index (JI) as a loss function. Let S be the inferred topics and T be the true topics. The Jaccard loss is defined as $\mathcal{L}_{\text{jaccard}}(S,T) = 1 - \frac{1}{k}\sum_{s \in S} \max_{t \in T} \frac{|\Gamma(s) \cap \Gamma(t)|}{|\Gamma(s) \cup \Gamma(t)|}$, where $k = |S| = |T|$ is the number of topics. When the clustering produced by the inferred and the true topics are similar, Jaccard loss is 0. When they are completely dissimilar, the loss is maximum, i.e., 1. Jaccard loss is a modular function.

3.4 Inference Algorithm: Greedy

Having learnt the weights for the mixture components, the resulting function $F_w(S) = \sum_{i=1}^{m} w_i f_i(S)$ is a submodular function. In the case when the individual components are themselves monotone (all our functions in fact are), $F_w(S)$ can be optimized by the accelerated greedy algorithm (Minoux, 1978). Thanks to submodularity, we can obtain near optimal solutions very efficiently. In case the functions are all monotone submodular, we can guarantee that the solution is within $1 - 1/e$ factor from the optimal solution.

4 Experimental Results

To validate our approach, we make use of Wikipedia category structure as a topic DAG and apply our technique to the task of automatic generation of Wikipedia disambiguation pages. We pre-processed the category graph to eliminate the cycles in order to make it a DAG. Each Wikipedia disambiguation page is manually created by Wikipedia editors by grouping a collection of Wikipedia articles into several groups. Each group is then assigned a name, which serves as a topic for the group. Typically, a disambiguation page segregates around 20-30 articles into 5-6 groups. Our goal is to measure how accurately we can recreate the groups for a disambiguation page and label them, given only the collection of articles mentioned in that disambiguation page (when actual groupings and labels are hidden).

4.1 Datasets

We parsed the contents of Wikipedia disambiguation pages and extracted disambiguation page names, article groups and group names. We collected about 8000 disambiguation pages that had at least four groups on them. Wikipedia category structure is used as the topic DAG. We eliminated few administrative categories such as "Hidden Categories", "Articles needing cleanup", and the like. The final DAG had about 1M topics and 3M links.

4.2 Evaluation Metrics

Every group of articles on the Wikipedia disambiguation page is assigned a name by the editors. Unfortunately, these names may not correspond to the Wikipedia category (topic) names. For example, one of the groups on the "Matrix" disambiguation page has a name "Business and government" and there is no Wikipedia category by that name. However, the group names generated by our (and baseline) method are from the Wikipedia categories (which forms our topic DAG). In addition, there can be multiple relevant names for a group. For example, a group on a disambiguation page may be called "Calculus", but an algorithm may rightly generate "Vector Calculus". Hence we cannot evaluate the accuracy of an algorithm just by matching the generated group names to those on the disambiguation page. To alleviate this problem, we adopt cluster-based evaluation metrics. We treat every group of articles generated by an algorithm under a topic for a disambiguation page as a cluster of articles. These are considered as *inferred* clusters for a disambiguation page. We compare them against the actual grouping of articles on the Wikipedia disambiguation page by treating those groups as *true* clusters. We can now adopt Jaccard Index, F1-measure, and NMI (Normalized Mutual Information) based cluster evaluation metrics described in (Manning et al., 2008). For each disambiguation page in the test set, we compute every metric score and then average it over all the disambiguation pages.

4.3 Methods Compared

We validated our approach by comparing against several baselines described below. We also compared two variations of our approach as described next. For each of these cases (baselines and two variations) we generated and compared the metrics (Jaccard Index, F1-measure and NMI) as described in the previous section.

KM_{docs}: K-Means algorithm run on articles as TF-IDF vectors of words. The number of clusters K is set to the number of true clusters on the Wikipedia disambiguation page.

$\text{KMed}_{\text{docs}}$: K-Medoids algorithm with articles as TF-IDF vectors of words. The number of clusters are set as in KM_{docs}.

$\text{KMed}_{\text{topics}}$: K-Medoids run on topics as TF-IDF vectors of words. The words for each topic is taken from the articles that are in the transitive cover of the topic.

LDA_{docs}: LDA algorithm with the number of topics set to the number of true clusters on the Wikipedia disambiguation page. Each article is then grouped under the highest probability topic.

SMML_{cov}: This is the submodular mixture learning case explained in section 3.1.5. Here we consider a mixture of all the submodular functions governing coverage, diversity, fidelity and QC functions. However, we exclude the similarity based functions described in section 3.1.2. Coverage based functions have a time complexity of $O(n)$ whereas similarity based functions are $O(n^2)$. By excluding similarity based functions, we can compare the quality of the results with and without $O(n^2)$ functions. We learn the mixture weights from the training set and use them during inference on the test set to subset K topics through the submodular maximization (Equation 1).

$\text{SMML}_{\text{cov+sim}}$: This case is similar to SMML_{cov} except that, we include similarity based submodular mixture components. This makes the inference time complexity $O(n^2)$.

We do not compare against HSLDA, PAM and few other techniques cited in the related work sections because they do not produce a subset of K summary topics — these are not directly comparable with our work.

4.4 Evaluation Results

We show that the submodular mixture learning and maximization approaches, i.e., SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other approaches in various metrics. In all these experiments, we performed 5 fold cross validation to learn the parameters from 80% of the disambiguation pages and evaluated on the rest of the 20%, in each fold.

In Figure 2a we summarize the results of the comparison of the methods mentioned above on Jaccard Index, F1 measure and NMI. Our proposed techniques SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ outperform other techniques consistently.

In Figures 2b and 2c we measure the number of test instances (i.e., disambiguation queries) in which each of the algorithms dominate (win) in evaluation metrics. In 60% of the disambiguation queries, SMML_{cov} and $\text{SMML}_{\text{cov+sim}}$ approaches

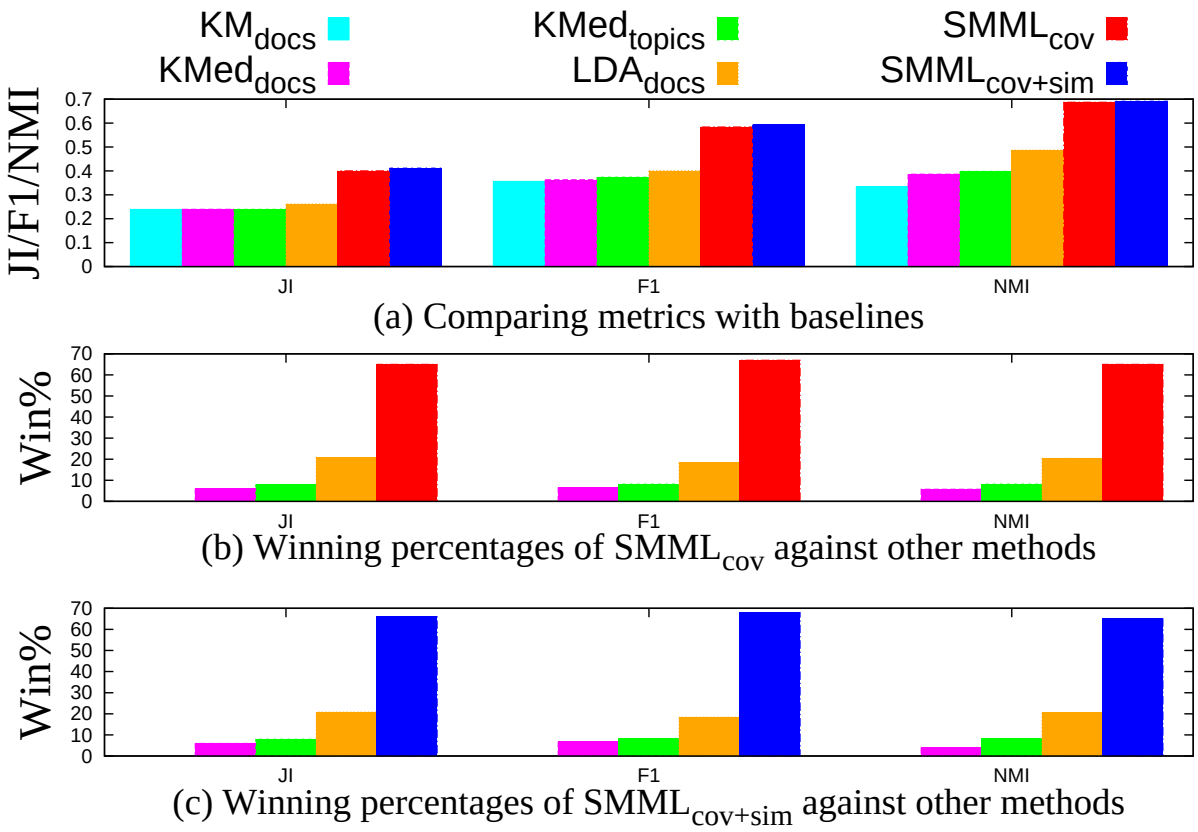

Figure 2: Comparison of techniques

produce higher JI, F1 and NMI than all other methods. This indicates that the clusters of articles produced by our technique resembles the clusters of articles present on the disambiguation page better than other techniques.

From Figures 2b and 2c it is clear that $O(n)$ time complexity based submodular mixture functions (SMML_{cov}) perform on par with $O(n^2)$ based functions ($\text{SMML}_{\text{cov+sim}}$), but at a greatly reduced execution time, demonstrating the sufficiency of $O(n)$ functions for our task. On the average, for each disambiguation query, SMML_{cov} took around 40 seconds (over 1M topics and 3M edges DAG) to infer the topics, whereas $\text{SMML}_{\text{cov+sim}}$ took around 35 minutes. Both these experiments were carried on a machine with 32 GB RAM, Eight-Core AMD Opteron(tm) Processor 2427.

5 Conclusions

We investigated a problem of summarizing topics over a massive topic DAG such that the summary set of topics produced represents the objects in the collection. This representation is characterized through various classes of submodular (and monotone) functions that captured coverage, similarity, diversity, specificity, clarity, relevance and fidelity of the topics. Currently we assume that the number of topics K is given as an input to our algorithm. It would be an interesting future problem to estimate the value of K automatically in our setting. As future work, we also plan to extend our techniques to produce a hierarchical summary of topics and scale it across heterogeneous collection of objects (from different domains) to bring all of them under the same topic DAG and investigate interesting cases thereon.

Acknowledgements: This material is based upon work supported by the National Science Foundation under Grant No. IIS-1162606, and by a Google, a Microsoft, and an Intel research award. Rishabh Iyer acknowledges support from the Microsoft Research Ph.D Fellowship.

References

Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. 2006. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, April.

W. Bi and J. T. Kwok. 2011. Multi-label classification on tree-and DAG-structured hierarchies. In *ICML*. ICML.

David M. Blei and John D. Lafferty. 2006. Correlated topic models. In *In Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. MIT Press.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

David M. Blei, Thomas L. Griffiths, Michael I. Jordan, and Joshua B. Tenenbaum. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*, page 2003. MIT Press.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Allan Borodin, Hyun Chul Lee, and Yuli Ye. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proceedings of Principles of Database Systems*, pages 155–166. ACM.

Razvan Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06), Trento, Italy*, pages 9–16, April.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE.

Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Sridhar Rajagopalan, Andrew Tomkins, John A. Tomlin, and Jason Y. Zien. 2003. Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 178–186, New York, NY, USA. ACM.

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA.

R. Iyer, S. Jegelka, and J. Bilmes. 2013. Fast semidifferential-based submodular function optimization. ICML.

Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, Angela Y. Wu, Senior Member, and Senior Member. 2002. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:881–892.

D. Kempe, J. Kleinberg, and E. Tardos. 2003. Maximizing the spread of influence through a social network. In *SIGKDD*.

Katrin Kirchhoff and Jeff Bilmes. 2014. Submodularity for data selection in machine translation. October.

A. Krause and C. Guestrin. 2005. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of Uncertainity in Artificial Intelligence*. UAI.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 577–584, New York, NY, USA. ACM.

H. Lin and J. Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL*.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *The 49th Meeting of the Assoc. for Comp. Ling. Human Lang. Technologies (ACL/HLT-2011)*, Portland, OR, June.

H. Lin and J. Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, page 479490.

Hui Lin. 2012. *Submodularity in Natural Language Processing: Algorithms and Applications*. Ph.D. thesis, University of Washington, Dept. of EE.

Min ling Zhang and Zhi hua Zhou. 2007. Ml-knn: A lazy learning approach to multi-label learning. *PATTERN RECOGNITION*, 40:2007.

Arun S. Maiya, John P. Thompson, Francisco Loaiza-Lemos, and Robert M. Rolfe. 2013. Exploratory analysis of highly heterogeneous document collections. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 1375–1383, New York, NY, USA. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.

Olena Medelyan, Ian H. Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the Wikipedia and AI workshop at AAAI-08*. AAAI.

Qiaozhu Mei, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 490–499, New York, NY, USA. ACM.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA. ACM.

David Milne. 2009. An open-source toolkit for mining wikipedia. In *In Proc. New Zealand Computer Science Research Student Conf*, page 2009.

Michel Minoux. 1978. Accelerated greedy algorithms for maximizing submodular set functions. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, chapter 27, pages 234–243. Springer Berlin Heidelberg, Berlin/Heidelberg.

Mukund Narasimhan and Jeff Bilmes. 2004. PAC-learning bounded tree-width graphical models. In *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference (UAI-2004)*. Morgan Kaufmann Publishers, July.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294.

Adler J. Perotte, Frank Wood, Noemie Elhadad, and Nicholas Bartlett. 2011. Hierarchically supervised latent dirichlet allocation. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *NIPS*, pages 2609–2617.

Juho Rousu, Craig Saunders, Sndor Szedmk, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626.

Jr. Silla, CarlosN. and AlexA. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA. ACM.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Sebastian Tschiatschek, Rishabh Iyer, Hoachen Wei, and Jeff Bilmes. 2014. Learning Mixtures of Submodular Functions for Image Collection Summarization. In *Neural Information Processing Systems (NIPS)*.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining multi-label data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer US.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2014a. Fast multi-stage submodular maximization. In *ICML*.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, Chris Bartels, and Jeff Bilmes. 2014b. Submodular subset selection for large-scale speech training data. *Proceedings of ICASSP, Florence, Italy*.

Torsten Zesch and Iryna Gurevych. 2007. Analysis of the wikipedia category graph for nlp applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, pages 1–8, Rochester, April. Association for Computational Linguistics.

BTP-I

Ilindra Sai Lakshmi Shreya(190050050)
Guide: Prof. Ganesh Ramakrishnan

Indian Institute of Technology Bombay

1 Study on DictDis

1.1 What is DictDis?

DictDis : Dictionary Constrained Disambiguation for Improved NMT is a lexically constrained NMT system that disambiguates between multiple candidate translations derived from dictionaries. Domain-specific neural machine translation can benefit highly from lexical constraints drawn from domain-specific dictionaries. Prior work has largely focused on the single candidate setting where the target word or phrase is replaced by a single constraint. But dictionaries could present multiple candidate translations for a source word/phrase on account of the polysemous nature of words. DictDis learns to disambiguate the multiple constraints that correspond to a single word/phrase.

The model is trained in a soft manner such that no constraint is forced to appear in the predicted sentence. Given a source sentence and domain specific dictionary constraints consisting of multiple source-target phrases, it can either pick the most relevant constraint or abstain from picking any constraint in order to least affect the fluency of the translation

1.2 Observations

1.2.1 Incomplete Augmented representation

Given a triplet of source sentence, constraints and target sentence (X, C, Y) where $C = \{C_i^j\}$ representing i^{th} inter-phrase constraint and j^{th} intra-phrase constraint, lexical constraints C are added into the pipeline by creating an augmented representation of source sentence and the constraints as follows

$$\hat{X} = [X, \langle sep \rangle, C_1^1, \langle isep \rangle, C_1^2, \langle sep \rangle, C_2^1, ..., C_n, \langle eos \rangle] \tag{1}$$

$\langle sep \rangle$, $\langle isep \rangle$ and $\langle eos \rangle$ are symbols indicating inter-phrase constraint separation, intra-phrase constraint separation and end of sentence respectively.

It leads to the loss of some key information that which source word/phrase does each of the constraints corresponding to. This correspondence could help very much in generating the translations. Therefore, using a representation that can capture this correspondence could help improve the model's performance.

1.2.2 Fanout-based Probability Distribution

Final distribution over target vocabulary is defined as the weighted sum of prediction probability and constraint ingestion probability as follows

$$p(y_t|(y_{<t}, \hat{X})) = g_t P_t^{pred} + (1 - g_t)(P_t^{copy} + P_t^{dis}) \tag{2}$$

P_t^{pred}, P_t^{copy} and P_t^{dis} are prediction probability, copy probability and disambiguate probability respectively.

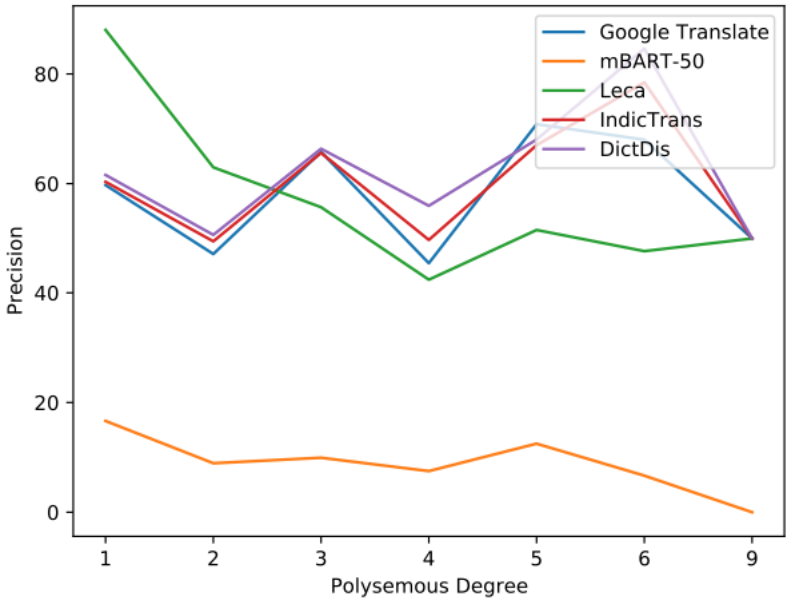

Fig. 1: SPDI for Aerospace dataset

Figures 1, 2, 3 show SPDI for various datasets obtained by various models including DictDis. Clearly the the performance of the model depends on the fanout of the constraints. Modelling fanout-based probability distribution could thererfore help improve the model's performance. One possible way could be adding fanout dependent weights. Currently work is being done in this direction, differentiating the cases of fanout =1 and fanout > 1 as follows

Disamb component :
Case 1 when $f_{t=1} : (1 - g_t)w(p_t^{copy} + p_t^{dis})$
Case 2 when $f_{t>1} : (1 - g_t)(1 - w)(p_t^{copy} + p_t^{dis})$

$$p(y_t|y_{<t}, X) = Disamb\ Component + g_t p_t^{predict} \tag{3}$$

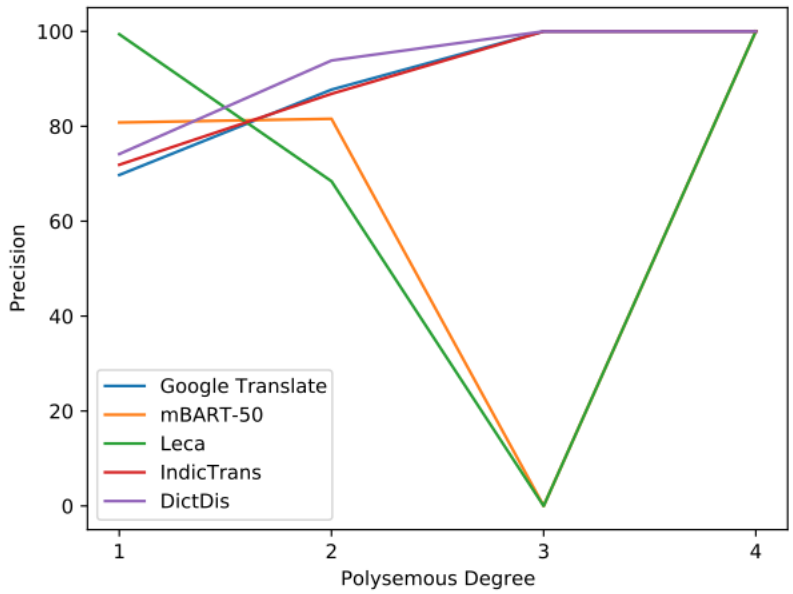

Fig. 2: SPDI for Banking dataset

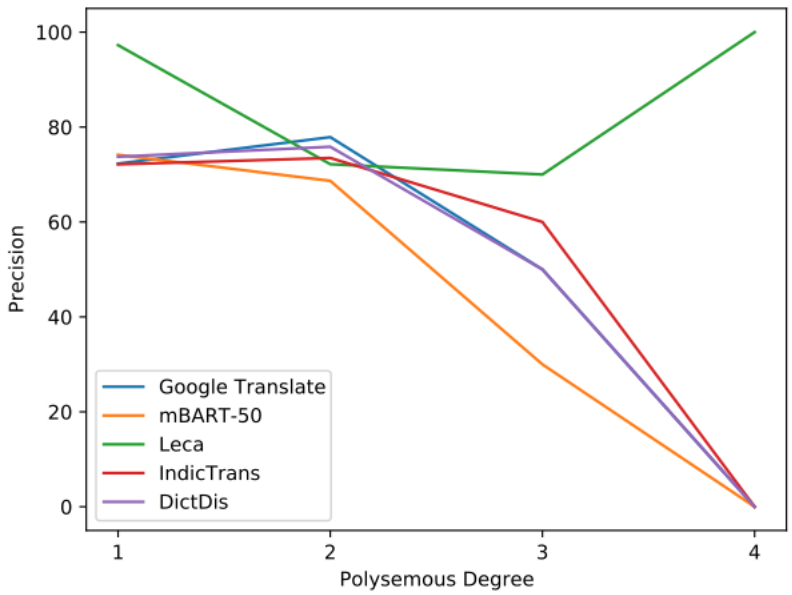

Fig. 3: SPDI for Regulatory dataset

1.2.3 Dictionary Usage by the Model

SPDI(figures 1, 2, 3) for lower fanout, especially at 1 in any domain is lower for DictDis compared to Leca. It is at the level of IndicTrans, a model which does not use any dictionary. Bleu(figure 4) scores for Banking and Regulatory domains are better for other models. These are the domains with less ambiguity i.e most of the constraints would have lower fanout.

	Regulatory	Aerospace	Banking	Combustion-T1	Combustion-T2	Structures	Medical
GoogleTrans	24.9	36	31.3	43.2	40.1	43	15.6
mBART	24.7	28.1	28.1	31.1	28.9	34.2	17
Leca	**28.4**	26.9	32.6	30.4	38.9	37.9	16.3
IndicTrans	27.3	37.3	**34.4**	43.3	41.3	47.2	15.7
DictDis	27.5	**37.9**	34	**44.9**	**43.4**	**51.3**	**20.6**

Fig. 4: BLEU scores for various datasets and approaches

These observations bring in the following questions

1. Whether dictionary is being used at all by the model?
   - Are the translations present in dictionary very common in the dataset that the model picks correct translation from the data itself rather than picking from dictionary?
2. Why the model is performing comparatively low when fanout =1?
   - For fanout 1 replacement(vs. no replacement) seems more aggressive on test sets relative to the replacement of fanout > 1, does it remain same for training sets?

To understand and answer these questions, we performed frequency analysis on the occurrence of dictionary words/phrases in the corpus.

2 Frequency Analysis of Dictionary Data in Samananthar Dataset

2.1 Problem Statement

Pondering the question why when fanout is 1, DictDis remained at the level of IndicTrans which does not use any dictionary. We want to check whether the model is getting to learn to use the dictionary and pick constraints from it at all. We check the frequency of occurrences of dictionary words/phrases in the training corpus. This would show where the problem exactly is

- Distribution and diversity of the dictionary dataset and training corpus used
- (or) The model itself

2.2 Processing

Following quantities are calculated (code) on and the training corpus, Samananthar and the generalised en-hi dictionary used for training

1. No. of LHS words/phrases of dictionary with non-zero occurrences in corpus
2. Frequency of LHS words/phrases in the source sentences (call it "LHS freq")
3. Frequency of LHS words/phrases in the source sentences when RHS translations occur in corresponding target sentences (call it "LRHS freq")

2.3 Results and Observations

2.3.1

Number of words/phrases in the dictionary = 11,767
Among these 11,593 have non-zero occurrences and 7,799 i.e approximately 75% have frequency greater than 100. Large fraction of the dictionary do appear in the source sentences. Hence, we can guarantee that learning to use the dictionary is possible from this dataset.

2.3.2

For the entire corpus and dictionary

LHS freq = 2,90,75,364
LRHS freq = 82,74,281
Constraint copy rate = 1 : 3 (approx)

i.e. for every 3 constraints given by the dictionary 1 constraint is copied into the translation in the training corpus. This is a fair distribution of constraints being copied. Therefore, the dictionary translations are not too common in the data. Model should be able to learn to use the dictionary with this data.

2.3.3

For the words/phrases with fanout = 1

LHS freq = 20,96,848
LRHS freq = 6,27,896
Constraint copy rate = 1 : 3.3 (approx)

For the words/phrases with fanout > 1
LHS freq = 2,69,78,516
LRHS freq = 76,46,385
Constraint copy rate = 1 : 3.5 (approx)

Constraint copying is equally aggressive for both the cases of fanout =1 and > 1, unlike test sets where ratios are even more skewed. This could partly explain the performance issues of fanout =1.
Word-by-word frequencies and analysis can be found here

3 Improving DictDis with Knowledge Distillation

We have kept some initial efforts in improving DictDis using knowledge-distillation methods by creating a student-teacher model. The teacher model would use dictionaries to add the lexical constraints and the student model learns how to disambiguate and select those constraints from the teacher. We wanted to follow the AMAL : Adaptive Mixing of Auxiliary Losses setting.

We have kept it on hold owing to few constraints. If we have a student model for learning disambiguation and copy part, then we have 2 large same-sized models for teacher and student. This makes the training quite time-taking. It also seemed unnecessary as the model itself is deep and copy part is being able learn here itself. Also the process of inference remains in questions as teacher model would be discarded that actually provides the constraints to disambiguate.

4 Lexicon Selection for Lexically Constrained NMT

4.1 Problem Statement

Currently user is asked to select the dictionaries to add lexical constraints from to the translations. We want to make this step automatic. Based on the source text, model should be able to infer the domain and accordingly select the relevant dictionaries to add constraints. Given a set of dictionaries and source text, we aim to select a minimal subset of dictionaries that covers most of the source text.

4.2 Approaches Summary

As a first step, we aim to create a baseline that involves no semantics i.e when we check for coverage of the source by a dictionary, we use simple syntactical token match after lemmatisation instead of matching semantically.

We do not expect constraints of common words to get added. Such words are of least importance and become noise when finding the coverage. To avoid this, we do the following

1. Remove stop words as a pre-processing step after tokenisation
2. Use idfs calculated on Samananthar corpus as weights while finding coverage

4.3 Weighted Overlap

4.3.1 Formulation

A similarity score is calculated for each dictionary. Using a threshold on this similarity score, the top dictionaries are give as the selected ones. This similarity score captures the coverage by calculating number of tokens in the source that occur in the dictionary weighted by idf of the token indicating its uniqueness of appearance.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ corpus}$$

For a phrasal match, we add number of words in phrase instead of single match.

4.3.2 Results

The model is run on several Wikipedia articles and its results (figure 5) are analysed qualitatively. A mixture of dictionaries are being selected - few domain related, general large sized(like chem, defence, eng-hin-gen) along with some noise. But domain-specific dictionaries are not at all picked up. For example, for the article on Dinosaur, zoology is getting selected but palaeontology isn't.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
electrical.csv	electrical.csv	electrical.csv	electrical.csv
chem.csv	nematology.csv	aerospace_glossary.csv	med_comp.csv
aerospace_glossary.csv	aerospace_glossary.csv	geography.csv	aerospace_glossary.csv
geography.csv	geophysics.csv	chem.csv	nematology.csv
nematology.csv	geography.csv	med_comp.csv	geophysics.csv
med_comp.csv	it.csv	it.csv	chem.csv
it.csv	itGlossary.csv	itGlossary.csv	it.csv
itGlossary.csv	med_comp.csv	nematology.csv	itGlossary.csv
geophysics.csv	broadcasting.csv	defence.csv	geography.csv
broadcasting.csv	chem.csv	geophysics.csv	zoology.csv
defence.csv	history.csv	broadcasting.csv	broadcasting.csv
mining_geo.csv	defence.csv	climatology.csv	med.csv
med.csv	itihas_paribhasha.csv	administrative.csv	clinical.csv

Fig. 5: Preference order of dictionaries for Wiki articles given by weighted overlap

4.4 Weighted Overlap with dictionary size normalization

4.4.1 Formulation

For some dictionaries, the size itself can be very large and general, due to which they might achieve undesirably higher similarity score in the previous formulation. To overcome this, a new formulation that performs normalization with dictionary size is used.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ dictionary}$$

4.4.2 Results

The results (figure 6) were very poor. The dictionary size normalisation seems to be overdoing. All the small sized dictionaries were pushed to top leaving no relevant dictionaries in the selection.

4.5 Observations and next steps

We found results of weighted overlap methods unsatisfying. Moreover, it does not model the set cover we desire. Set cover implies subset of dictionaries that together try to cover the entire source. Unlike, weighted overlap

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv
adhoc.csv	geophysics.csv	geophysics.csv	geophysics.csv
geophysics.csv	nematology.csv	nematology.csv	nematology.csv
nematology.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
eng-hindi.csv	adhoc.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi-general.csv	eng-hindi-general.csv	it.csv	it.csv
it.csv	it.csv	itGlossary.csv	itGlossary.csv
itGlossary.csv	itGlossary.csv	broadcasting.csv	broadcasting.csv
broadcasting.csv	itihas_paribhasha.csv	NGMA_Dict.csv	palaeobotany.csv
western_music.csv	broadcasting.csv	maanachitravijnan.csv	western_music.csv
NGMA_Dict.csv	NGMA_Dict.csv	western_music.csv	adhoc.csv
palaeobotany.csv	sociology.csv	kosh_vigyan.csv	NGMA_Dict.csv
agri.csv	western_music.csv	phy.csv	palaeontology.csv
economic_geo.csv	history.csv	physicsGlossary.csv	phy_mal.csv
kosh_vigyan.csv	palaeobotany.csv	administrative.csv	phy.csv

Fig. 6: Preference order of dictionaries for Wiki articles given by weighted overlap with dictionary size normalisation

- Finds amount of overlap of a dictionary independent of the other dictionaries
- Selection of a subset of dictionaries is done based on a given threshold i.e. a minimum amount of overlap required which would give the best dictionaries in terms of "a dictionary alone being able to represent the source"

We hence moved to use the subset selecting sub-modlar functions provided by submidlib. They assign a score for a subset. Based on a given budget i.e. size of subset, the optimal one is found.

4.6 Problem with Graph-cut

For covering a set U, by a subset X of V, Graph-cut function is defined as

$$f_{gc}(X) = \sum_{i \in U, j \in X} s_{ij} - \lambda \sum_{i,j \in X} s_{ij} \tag{4}$$

Matching it to our problem setting

- U is the source text, then i could be modeled as a token
- X is the subset of dictionaries, then j is a dictionary in the subset

Hence, s_{ij} is similarity between a word and an entire dictionary which is not we want to model in our problem. Therefore graph-cut is not the right choice.

4.7 Set Cover

4.7.1 Formulation

It is an implementation of the regular set cover. For covering a set of concepts U, by a subset A of V, its Set Cover evaluation is defined as

$$f(A) = w(\cup_{a \in A} \gamma(a)) = w(\gamma(A)) \tag{5}$$

where $\gamma(A)$ refers to the set of concepts covered by A and $w(\gamma(A))$ is total weight of concepts covered by elements in A

Matching it to our problem setting, the concepts to be covered are tokens in the source, A is the subset of dictionaires, $\gamma(a)$ is the number of tokens of source that occur in dictionary a and w is the idf weighted coverage.

4.7.2 Results

The results (figure 7) are satisfying for a baseline. Domain specific dictionaries are coming in the top. For example maanachitravijnan for Topographic map. General large sized dictionaries(like chem, defence, eng-hin-gen) are coming along with due to their large coverage of the source.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
mining_geo.csv	geography.csv	geography.csv	zoology.csv
chem.csv	electrical.csv	sociology.csv	geography.csv
geography.csv	defence.csv	physicsGlossary.csv	palaeobotany.csv
climatology.csv	med_comp.csv	maanachitravijnan.csv	med_comp.csv
petrology.csv	petrology.csv	chem.csv	chem.csv
sociology.csv	sociology.csv	geology.csv	defence.csv
med_comp.csv	hom_sci.csv	broadcasting.csv	history.csv
geology.csv	itihas_paribhasha.csv	linguistics.csv	bio.csv
bio.csv	electronics.csv	history.csv	botany.csv
mineralogy.csv	lib_info.csv	chem_eng.csv	sociology.csv
chem_eng.csv	clinical.csv	defence.csv	clinical.csv
palaeontology.csv	zoology.csv	psychology.csv	hom_sci.csv
history.csv	quality_cont.csv	med_comp.csv	palaeontology.csv
linguistics.csv	history.csv	bio.csv	electronics.csv

Fig. 7: Preference order of dictionaries for Wiki articles given by Set Cover

Code for all the above implementations can be found here

4.8 Deployment

The SetCover implementation is deployed on the AICTE interfaces (1, 2 and 3)

A challenge in the process of deploying is the position of dictionary selection in the entire pipeline. The selection is to be performed after OCR. Current OCR is highly sophisticated and takes considerable amount of time for large books. Hence providing a preference order of dictionaries and then asking the user to perform selection requires user to intervene in the process after a considerable time.

To overcome this issue, as a current workaround we ask user to give number of dictionaries to auto-select and use them automatically further down the pipeline.

Work is being done on giving user a preference order of dictionaries to select from. This we currently add only for the machine-readable sources on which simple-quick OCR is done to obtain the text.

Code can be found here

5 Future Work

5.1 Extension for multilingual NMT

Current implementation performs pre-processing considering the source language is english. This has to be generalised by performing language-specific pre-processing

Currently subsetting is done from all the available dictionaries. In case of multilingual NMT, there would be dictionaries of multiple source and translation languages. Subsetting should be done only from those dictionaried corresponding to the specified source and translation languages.

5.2 Analysis of current Implementation

Setup profiling feedback on the interface for dictionary selection. Come up with measures that can quantify the results of this functionality.

5.3 Setting up a Learning Problem

Develop the model as a learning problem where we learn the weights of a mixture of multiple submodular and modular functions using max margin as in here.

We seem to need a mixture of different submodular set coverage functions (with normalization, without normalization, with idf, without idf) and several modular functions (quality of each dicitonary, quality of the words being covered - say nouns vs. verbs) and learning mixtures of those

5.4 Keyword based coverage

Translating books is one of the major aims of the pipeline. Books are a structured data, with contents and keywords providing the major information about it. Hence, finding coverage based on just contents/titles/sections/keywords instead of the entire book could work well.

This idea can be incorporated separately to make the subset selection fast. It can also be included in the multiple mixture components in the learning setting.

BTP-I

Ilindra Sai Lakshmi Shreya(190050050)
Guide: Prof. Ganesh Ramakrishnan

Indian Institute of Technology Bombay

1 Study on DictDis

1.1 What is DictDis?

DictDis : Dictionary Constrained Disambiguation for Improved NMT is a lexically constrained NMT system that disambiguates between multiple candidate translations derived from dictionaries. Domain-specific neural machine translation can benefit highly from lexical constraints drawn from domain-specific dictionaries. Prior work has largely focused on the single candidate setting where the target word or phrase is replaced by a single constraint. But dictionaries could present multiple candidate translations for a source word/phrase on account of the polysemous nature of words. DictDis learns to disambiguate the multiple constraints that correspond to a single word/phrase.

The model is trained in a soft manner such that no constraint is forced to appear in the predicted sentence. Given a source sentence and domain specific dictionary constraints consisting of multiple source-target phrases, it can either pick the most relevant constraint or abstain from picking any constraint in order to least affect the fluency of the translation

1.2 Observations

1.2.1 Incomplete Augmented representation

Given a triplet of source sentence, constraints and target sentence (X, C, Y) where $C = \{C_i^j\}$ representing i^{th} inter-phrase constraint and j^{th} intra-phrase constraint, lexical constraints C are added into the pipeline by creating an augmented representation of source sentence and the constraints as follows

$$\hat{X} = [X, \langle sep \rangle, C_1^1, \langle isep \rangle, C_1^2, \langle sep \rangle, C_2^1, ..., C_n, \langle eos \rangle] \tag{1}$$

$\langle sep \rangle$, $\langle isep \rangle$ and $\langle eos \rangle$ are symbols indicating inter-phrase constraint separation, intra-phrase constraint separation and end of sentence respectively.

It leads to the loss of some key information that which source word/phrase does each of the constraints corresponding to. This correspondence could help very much in generating the translations. Therefore, using a representation that can capture this correspondence could help improve the model's performance.

1.2.2 Fanout-based Probability Distribution

Final distribution over target vocabulary is defined as the weighted sum of prediction probability and constraint ingestion probability as follows

$$p(y_t|(y_{<t}, \hat{X})) = g_t P_t^{pred} + (1 - g_t)(P_t^{copy} + P_t^{dis}) \tag{2}$$

P_t^{pred}, P_t^{copy} and P_t^{dis} are prediction probability, copy probability and disambiguate probability respectively.

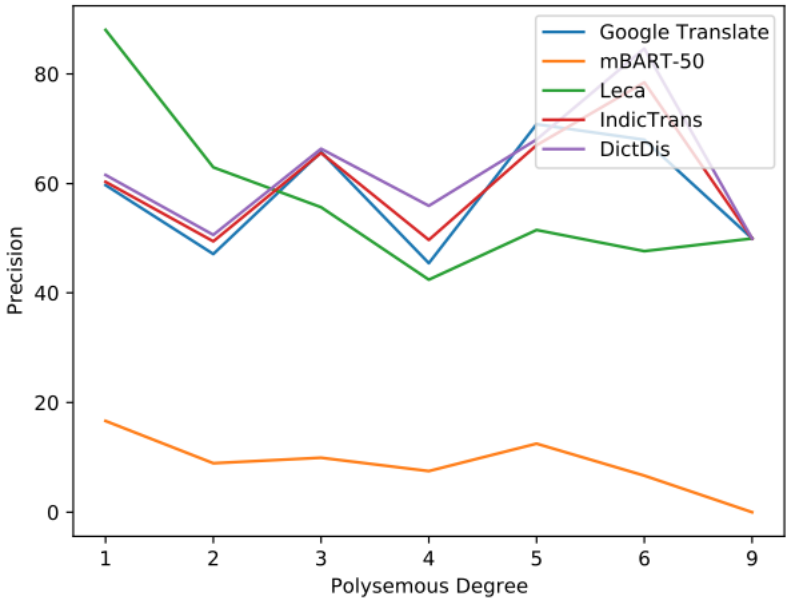

Fig. 1: SPDI for Aerospace dataset

Figures 1, 2, 3 show SPDI for various datasets obtained by various models including DictDis. Clearly the the performance of the model depends on the fanout of the constraints. Modelling fanout-based probability distribution could thererfore help improve the model's performance. One possible way could be adding fanout dependent weights. Currently work is being done in this direction, differentiating the cases of fanout =1 and fanout > 1 as follows

Disamb component :
Case 1 when $f_{t=1} : (1 - g_t)w(p_t^{copy} + p_t^{dis})$
Case 2 when $f_{t>1} : (1 - g_t)(1 - w)(p_t^{copy} + p_t^{dis})$

$$p(y_t|y_{<t}, X) = Disamb\ Component + g_t p_t^{predict} \tag{3}$$

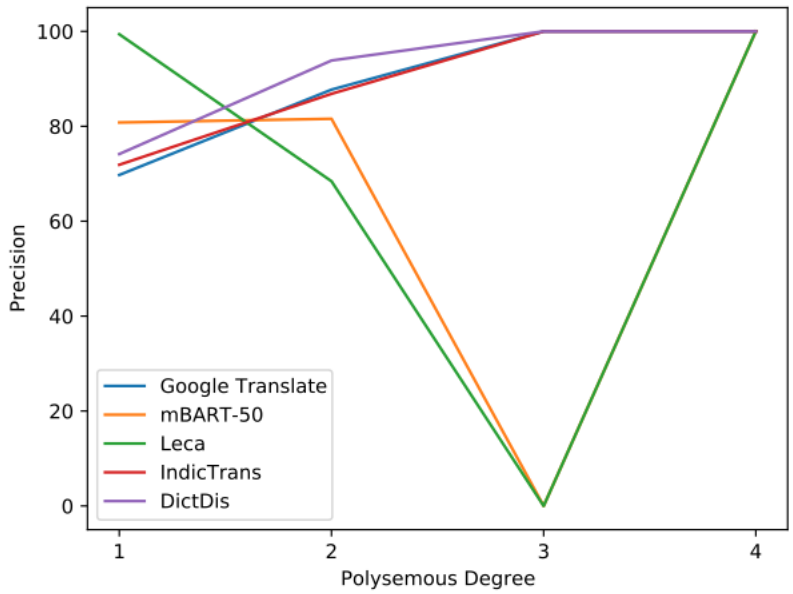

Fig. 2: SPDI for Banking dataset

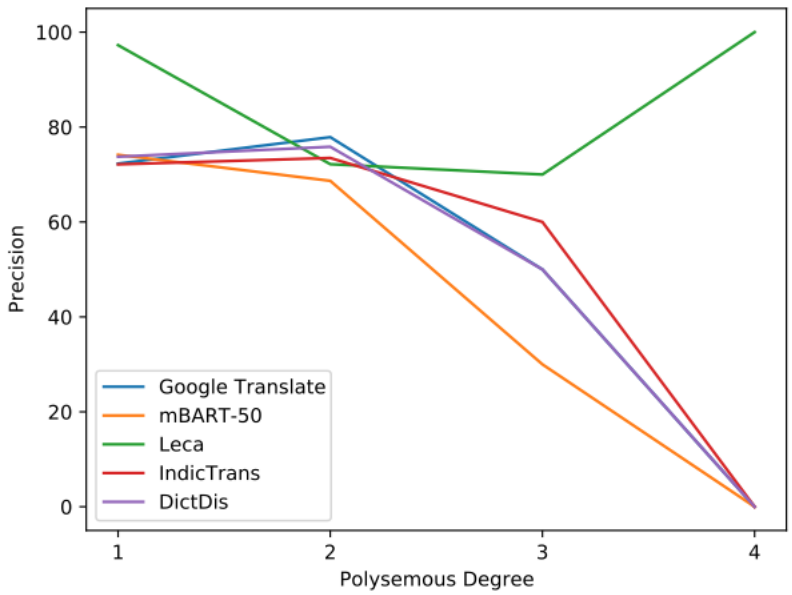

Fig. 3: SPDI for Regulatory dataset

1.2.3 Dictionary Usage by the Model

SPDI(figures 1, 2, 3) for lower fanout, especially at 1 in any domain is lower for DictDis compared to Leca. It is at the level of IndicTrans, a model which does not use any dictionary. Bleu(figure 4) scores for Banking and Regulatory domains are better for other models. These are the domains with less ambiguity i.e most of the constraints would have lower fanout.

	Regulatory	Aerospace	Banking	Combustion-T1	Combustion-T2	Structures	Medical
GoogleTrans	24.9	36	31.3	43.2	40.1	43	15.6
mBART	24.7	28.1	28.1	31.1	28.9	34.2	17
Leca	**28.4**	26.9	32.6	30.4	38.9	37.9	16.3
IndicTrans	27.3	37.3	**34.4**	43.3	41.3	47.2	15.7
DictDis	27.5	**37.9**	34	**44.9**	**43.4**	**51.3**	**20.6**

Fig. 4: BLEU scores for various datasets and approaches

These observations bring in the following questions

1. Whether dictionary is being used at all by the model?
   - Are the translations present in dictionary very common in the dataset that the model picks correct translation from the data itself rather than picking from dictionary?
2. Why the model is performing comparatively low when fanout =1?
   - For fanout 1 replacement(vs. no replacement) seems more aggressive on test sets relative to the replacement of fanout > 1, does it remain same for training sets?

To understand and answer these questions, we performed frequency analysis on the occurrence of dictionary words/phrases in the corpus.

2 Frequency Analysis of Dictionary Data in Samananthar Dataset

2.1 Problem Statement

Pondering the question why when fanout is 1, DictDis remained at the level of IndicTrans which does not use any dictionary. We want to check whether the model is getting to learn to use the dictionary and pick constraints from it at all. We check the frequency of occurrences of dictionary words/phrases in the training corpus. This would show where the problem exactly is

- Distribution and diversity of the dictionary dataset and training corpus used
- (or) The model itself

2.2 Processing

Following quantities are calculated (code) on and the training corpus, Samananthar and the generalised en-hi dictionary used for training

1. No. of LHS words/phrases of dictionary with non-zero occurrences in corpus
2. Frequency of LHS words/phrases in the source sentences (call it "LHS freq")
3. Frequency of LHS words/phrases in the source sentences when RHS translations occur in corresponding target sentences (call it "LRHS freq")

2.3 Results and Observations

2.3.1

Number of words/phrases in the dictionary = 11,767
Among these 11,593 have non-zero occurrences and 7,799 i.e approximately 75% have frequency greater than 100. Large fraction of the dictionary do appear in the source sentences. Hence, we can guarantee that learning to use the dictionary is possible from this dataset.

2.3.2

For the entire corpus and dictionary

LHS freq = 2,90,75,364
LRHS freq = 82,74,281
Constraint copy rate = 1 : 3 (approx)

i.e. for every 3 constraints given by the dictionary 1 constraint is copied into the translation in the training corpus. This is a fair distribution of constraints being copied. Therefore, the dictionary translations are not too common in the data. Model should be able to learn to use the dictionary with this data.

2.3.3

For the words/phrases with fanout = 1

LHS freq = 20,96,848
LRHS freq = 6,27,896
Constraint copy rate = 1 : 3.3 (approx)

For the words/phrases with fanout > 1
LHS freq = 2,69,78,516
LRHS freq = 76,46,385
Constraint copy rate = 1 : 3.5 (approx)

Constraint copying is equally aggressive for both the cases of fanout =1 and > 1, unlike test sets where ratios are even more skewed. This could partly explain the performance issues of fanout =1.
Word-by-word frequencies and analysis can be found here

3 Improving DictDis with Knowledge Distillation

We have kept some initial efforts in improving DictDis using knowledge-distillation methods by creating a student-teacher model. The teacher model would use dictionaries to add the lexical constraints and the student model learns how to disambiguate and select those constraints from the teacher. We wanted to follow the AMAL : Adaptive Mixing of Auxiliary Losses setting.

We have kept it on hold owing to few constraints. If we have a student model for learning disambiguation and copy part, then we have 2 large same-sized models for teacher and student. This makes the training quite time-taking. It also seemed unnecessary as the model itself is deep and copy part is being able learn here itself. Also the process of inference remains in questions as teacher model would be discarded that actually provides the constraints to disambiguate.

4 Lexicon Selection for Lexically Constrained NMT

4.1 Problem Statement

Currently user is asked to select the dictionaries to add lexical constraints from to the translations. We want to make this step automatic. Based on the source text, model should be able to infer the domain and accordingly select the relevant dictionaries to add constraints. Given a set of dictionaries and source text, we aim to select a minimal subset of dictionaries that covers most of the source text.

4.2 Approaches Summary

As a first step, we aim to create a baseline that involves no semantics i.e when we check for coverage of the source by a dictionary, we use simple syntactical token match after lemmatisation instead of matching semantically.

We do not expect constraints of common words to get added. Such words are of least importance and become noise when finding the coverage. To avoid this, we do the following

1. Remove stop words as a pre-processing step after tokenisation
2. Use idfs calculated on Samananthar corpus as weights while finding coverage

4.3 Weighted Overlap

4.3.1 Formulation

A similarity score is calculated for each dictionary. Using a threshold on this similarity score, the top dictionaries are give as the selected ones. This similarity score captures the coverage by calculating number of tokens in the source that occur in the dictionary weighted by idf of the token indicating its uniqueness of appearance.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ corpus}$$

For a phrasal match, we add number of words in phrase instead of single match.

4.3.2 Results

The model is run on several Wikipedia articles and its results (figure 5) are analysed qualitatively. A mixture of dictionaries are being selected - few domain related, general large sized(like chem, defence, eng-hin-gen) along with some noise. But domain-specific dictionaries are not at all picked up. For example, for the article on Dinosaur, zoology is getting selected but palaeontology isn't.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
electrical.csv	electrical.csv	electrical.csv	electrical.csv
chem.csv	nematology.csv	aerospace_glossary.csv	med_comp.csv
aerospace_glossary.csv	aerospace_glossary.csv	geography.csv	aerospace_glossary.csv
geography.csv	geophysics.csv	chem.csv	nematology.csv
nematology.csv	geography.csv	med_comp.csv	geophysics.csv
med_comp.csv	it.csv	it.csv	chem.csv
it.csv	itGlossary.csv	itGlossary.csv	it.csv
itGlossary.csv	med_comp.csv	nematology.csv	itGlossary.csv
geophysics.csv	broadcasting.csv	defence.csv	geography.csv
broadcasting.csv	chem.csv	geophysics.csv	zoology.csv
defence.csv	history.csv	broadcasting.csv	broadcasting.csv
mining_geo.csv	defence.csv	climatology.csv	med.csv
med.csv	itihas_paribhasha.csv	administrative.csv	clinical.csv

Fig. 5: Preference order of dictionaries for Wiki articles given by weighted overlap

4.4 Weighted Overlap with dictionary size normalization

4.4.1 Formulation

For some dictionaries, the size itself can be very large and general, due to which they might achieve undesirably higher similarity score in the previous formulation. To overcome this, a new formulation that performs normalization with dictionary size is used.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ dictionary}$$

4.4.2 Results

The results (figure 6) were very poor. The dictionary size normalisation seems to be overdoing. All the small sized dictionaries were pushed to top leaving no relevant dictionaries in the selection.

4.5 Observations and next steps

We found results of weighted overlap methods unsatisfying. Moreover, it does not model the set cover we desire. Set cover implies subset of dictionaries that together try to cover the entire source. Unlike, weighted overlap

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv
adhoc.csv	geophysics.csv	geophysics.csv	geophysics.csv
geophysics.csv	nematology.csv	nematology.csv	nematology.csv
nematology.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
eng-hindi.csv	adhoc.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi-general.csv	eng-hindi-general.csv	it.csv	it.csv
it.csv	it.csv	itGlossary.csv	itGlossary.csv
itGlossary.csv	itGlossary.csv	broadcasting.csv	broadcasting.csv
broadcasting.csv	itihas_paribhasha.csv	NGMA_Dict.csv	palaeobotany.csv
western_music.csv	broadcasting.csv	maanachitravijnan.csv	western_music.csv
NGMA_Dict.csv	NGMA_Dict.csv	western_music.csv	adhoc.csv
palaeobotany.csv	sociology.csv	kosh_vigyan.csv	NGMA_Dict.csv
agri.csv	western_music.csv	phy.csv	palaeontology.csv
economic_geo.csv	history.csv	physicsGlossary.csv	phy_mal.csv
kosh_vigyan.csv	palaeobotany.csv	administrative.csv	phy.csv

Fig. 6: Preference order of dictionaries for Wiki articles given by weighted overlap with dictionary size normalisation

- Finds amount of overlap of a dictionary independent of the other dictionaries
- Selection of a subset of dictionaries is done based on a given threshold i.e. a minimum amount of overlap required which would give the best dictionaries in terms of "a dictionary alone being able to represent the source"

We hence moved to use the subset selecting sub-modlar functions provided by submidlib. They assign a score for a subset. Based on a given budget i.e. size of subset, the optimal one is found.

4.6 Problem with Graph-cut

For covering a set U, by a subset X of V, Graph-cut function is defined as

$$f_{gc}(X) = \sum_{i \in U, j \in X} s_{ij} - \lambda \sum_{i,j \in X} s_{ij} \tag{4}$$

Matching it to our problem setting

- U is the source text, then i could be modeled as a token
- X is the subset of dictionaries, then j is a dictionary in the subset

Hence, s_{ij} is similarity between a word and an entire dictionary which is not we want to model in our problem. Therefore graph-cut is not the right choice.

4.7 Set Cover

4.7.1 Formulation

It is an implementation of the regular set cover. For covering a set of concepts U, by a subset A of V, its Set Cover evaluation is defined as

$$f(A) = w(\cup_{a \in A} \gamma(a)) = w(\gamma(A)) \tag{5}$$

where $\gamma(A)$ refers to the set of concepts covered by A and $w(\gamma(A))$ is total weight of concepts covered by elements in A

Matching it to our problem setting, the concepts to be covered are tokens in the source, A is the subset of dictionaires, $\gamma(a)$ is the number of tokens of source that occur in dictionary a and w is the idf weighted coverage.

4.7.2 Results

The results (figure 7) are satisfying for a baseline. Domain specific dictionaries are coming in the top. For example maanachitravijnan for Topographic map. General large sized dictionaries(like chem, defence, eng-hin-gen) are coming along with due to their large coverage of the source.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
mining_geo.csv	geography.csv	geography.csv	zoology.csv
chem.csv	electrical.csv	sociology.csv	geography.csv
geography.csv	defence.csv	physicsGlossary.csv	palaeobotany.csv
climatology.csv	med_comp.csv	maanachitravijnan.csv	med_comp.csv
petrology.csv	petrology.csv	chem.csv	chem.csv
sociology.csv	sociology.csv	geology.csv	defence.csv
med_comp.csv	hom_sci.csv	broadcasting.csv	history.csv
geology.csv	itihas_paribhasha.csv	linguistics.csv	bio.csv
bio.csv	electronics.csv	history.csv	botany.csv
mineralogy.csv	lib_info.csv	chem_eng.csv	sociology.csv
chem_eng.csv	clinical.csv	defence.csv	clinical.csv
palaeontology.csv	zoology.csv	psychology.csv	hom_sci.csv
history.csv	quality_cont.csv	med_comp.csv	palaeontology.csv
linguistics.csv	history.csv	bio.csv	electronics.csv

Fig. 7: Preference order of dictionaries for Wiki articles given by Set Cover

Code for all the above implementations can be found here

4.8 Deployment

The SetCover implementation is deployed on the AICTE interfaces (1, 2 and 3)

A challenge in the process of deploying is the position of dictionary selection in the entire pipeline. The selection is to be performed after OCR. Current OCR is highly sophisticated and takes considerable amount of time for large books. Hence providing a preference order of dictionaries and then asking the user to perform selection requires user to intervene in the process after a considerable time.

To overcome this issue, as a current workaround we ask user to give number of dictionaries to auto-select and use them automatically further down the pipeline.

Work is being done on giving user a preference order of dictionaries to select from. This we currently add only for the machine-readable sources on which simple-quick OCR is done to obtain the text.

Code can be found here

5 Future Work

5.1 Extension for multilingual NMT

Current implementation performs pre-processing considering the source language is english. This has to be generalised by performing language-specific pre-processing

Currently subsetting is done from all the available dictionaries. In case of multilingual NMT, there would be dictionaries of multiple source and translation languages. Subsetting should be done only from those dictionaried corresponding to the specified source and translation languages.

5.2 Analysis of current Implementation

Setup profiling feedback on the interface for dictionary selection. Come up with measures that can quantify the results of this functionality.

5.3 Setting up a Learning Problem

Develop the model as a learning problem where we learn the weights of a mixture of multiple submodular and modular functions using max margin as in here.

We seem to need a mixture of different submodular set coverage functions (with normalization, without normalization, with idf, without idf) and several modular functions (quality of each dicitonary, quality of the words being covered - say nouns vs. verbs) and learning mixtures of those

5.4 Keyword based coverage

Translating books is one of the major aims of the pipeline. Books are a structured data, with contents and keywords providing the major information about it. Hence, finding coverage based on just contents/titles/sections/keywords instead of the entire book could work well.

This idea can be incorporated separately to make the subset selection fast. It can also be included in the multiple mixture components in the learning setting.

BTP-I

Ilindra Sai Lakshmi Shreya(190050050)
Guide: Prof. Ganesh Ramakrishnan

Indian Institute of Technology Bombay

1 Study on DictDis

1.1 What is DictDis?

DictDis : Dictionary Constrained Disambiguation for Improved NMT is a lexically constrained NMT system that disambiguates between multiple candidate translations derived from dictionaries. Domain-specific neural machine translation can benefit highly from lexical constraints drawn from domain-specific dictionaries. Prior work has largely focused on the single candidate setting where the target word or phrase is replaced by a single constraint. But dictionaries could present multiple candidate translations for a source word/phrase on account of the polysemous nature of words. DictDis learns to disambiguate the multiple constraints that correspond to a single word/phrase.

The model is trained in a soft manner such that no constraint is forced to appear in the predicted sentence. Given a source sentence and domain specific dictionary constraints consisting of multiple source-target phrases, it can either pick the most relevant constraint or abstain from picking any constraint in order to least affect the fluency of the translation

1.2 Observations

1.2.1 Incomplete Augmented representation

Given a triplet of source sentence, constraints and target sentence (X, C, Y) where $C = \{C_i^j\}$ representing i^{th} inter-phrase constraint and j^{th} intra-phrase constraint, lexical constraints C are added into the pipeline by creating an augmented representation of source sentence and the constraints as follows

$$\hat{X} = [X, \langle sep \rangle, C_1^1, \langle isep \rangle, C_1^2, \langle sep \rangle, C_2^1, ..., C_n, \langle eos \rangle] \tag{1}$$

$\langle sep \rangle$, $\langle isep \rangle$ and $\langle eos \rangle$ are symbols indicating inter-phrase constraint separation, intra-phrase constraint separation and end of sentence respectively.

It leads to the loss of some key information that which source word/phrase does each of the constraints corresponding to. This correspondence could help very much in generating the translations. Therefore, using a representation that can capture this correspondence could help improve the model's performance.

1.2.2 Fanout-based Probability Distribution

Final distribution over target vocabulary is defined as the weighted sum of prediction probability and constraint ingestion probability as follows

$$p(y_t|(y_{<t}, \hat{X})) = g_t P_t^{pred} + (1 - g_t)(P_t^{copy} + P_t^{dis}) \tag{2}$$

P_t^{pred}, P_t^{copy} and P_t^{dis} are prediction probability, copy probability and disambiguate probability respectively.

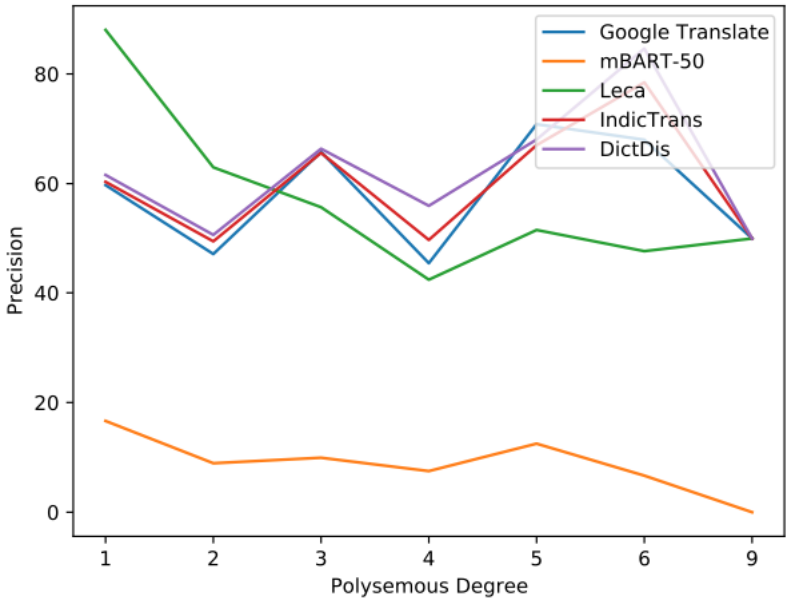

Fig. 1: SPDI for Aerospace dataset

Figures 1, 2, 3 show SPDI for various datasets obtained by various models including DictDis. Clearly the the performance of the model depends on the fanout of the constraints. Modelling fanout-based probability distribution could thererfore help improve the model's performance. One possible way could be adding fanout dependent weights. Currently work is being done in this direction, differentiating the cases of fanout =1 and fanout > 1 as follows

Disamb component :
Case 1 when $f_{t=1} : (1 - g_t)w(p_t^{copy} + p_t^{dis})$
Case 2 when $f_{t>1} : (1 - g_t)(1 - w)(p_t^{copy} + p_t^{dis})$

$$p(y_t|y_{<t}, X) = Disamb\ Component + g_t p_t^{predict} \tag{3}$$

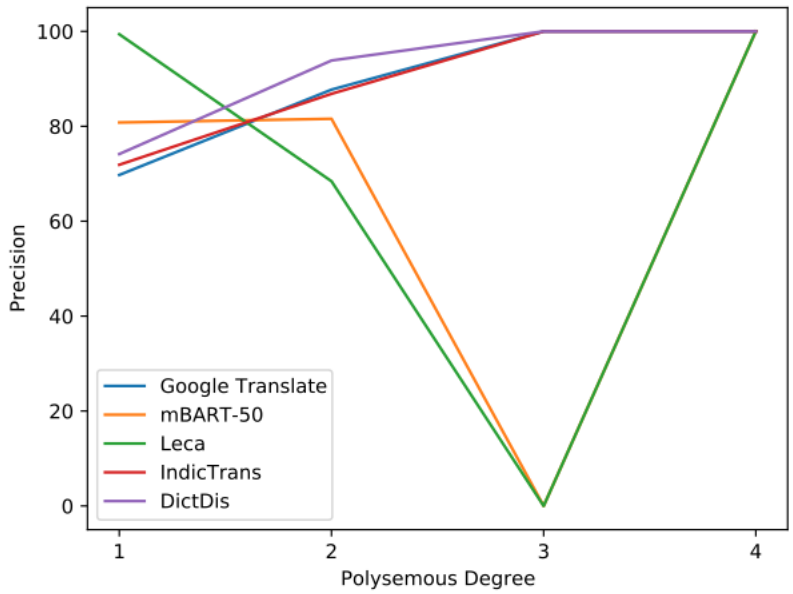

Fig. 2: SPDI for Banking dataset

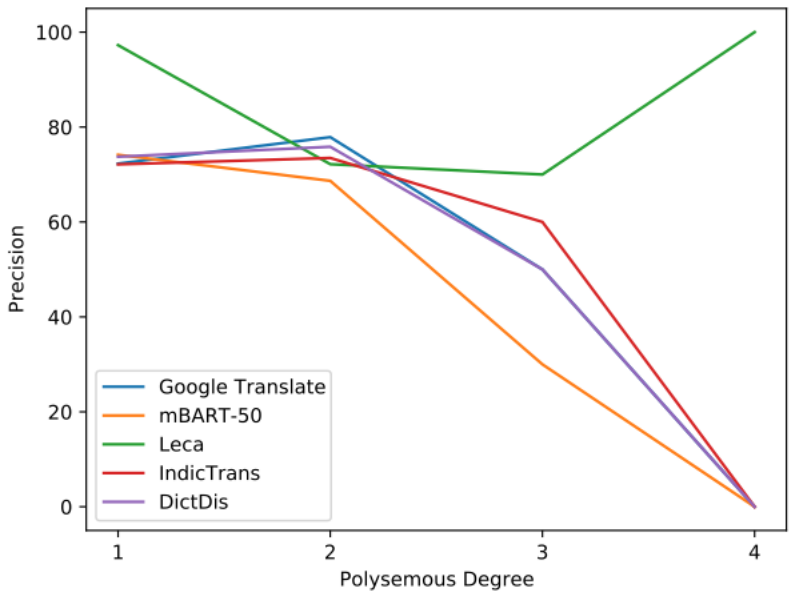

Fig. 3: SPDI for Regulatory dataset

1.2.3 Dictionary Usage by the Model

SPDI(figures 1, 2, 3) for lower fanout, especially at 1 in any domain is lower for DictDis compared to Leca. It is at the level of IndicTrans, a model which does not use any dictionary. Bleu(figure 4) scores for Banking and Regulatory domains are better for other models. These are the domains with less ambiguity i.e most of the constraints would have lower fanout.

	Regulatory	Aerospace	Banking	Combustion-T1	Combustion-T2	Structures	Medical
GoogleTrans	24.9	36	31.3	43.2	40.1	43	15.6
mBART	24.7	28.1	28.1	31.1	28.9	34.2	17
Leca	**28.4**	26.9	32.6	30.4	38.9	37.9	16.3
IndicTrans	27.3	37.3	**34.4**	43.3	41.3	47.2	15.7
DictDis	27.5	**37.9**	34	**44.9**	**43.4**	**51.3**	**20.6**

Fig. 4: BLEU scores for various datasets and approaches

These observations bring in the following questions

1. Whether dictionary is being used at all by the model?
   - Are the translations present in dictionary very common in the dataset that the model picks correct translation from the data itself rather than picking from dictionary?
2. Why the model is performing comparatively low when fanout =1?
   - For fanout 1 replacement(vs. no replacement) seems more aggressive on test sets relative to the replacement of fanout > 1, does it remain same for training sets?

To understand and answer these questions, we performed frequency analysis on the occurrence of dictionary words/phrases in the corpus.

2 Frequency Analysis of Dictionary Data in Samananthar Dataset

2.1 Problem Statement

Pondering the question why when fanout is 1, DictDis remained at the level of IndicTrans which does not use any dictionary. We want to check whether the model is getting to learn to use the dictionary and pick constraints from it at all. We check the frequency of occurrences of dictionary words/phrases in the training corpus. This would show where the problem exactly is

- Distribution and diversity of the dictionary dataset and training corpus used
- (or) The model itself

2.2 Processing

Following quantities are calculated (code) on and the training corpus, Samananthar and the generalised en-hi dictionary used for training

1. No. of LHS words/phrases of dictionary with non-zero occurrences in corpus
2. Frequency of LHS words/phrases in the source sentences (call it "LHS freq")
3. Frequency of LHS words/phrases in the source sentences when RHS translations occur in corresponding target sentences (call it "LRHS freq")

2.3 Results and Observations

2.3.1

Number of words/phrases in the dictionary = 11,767
Among these 11,593 have non-zero occurrences and 7,799 i.e approximately 75% have frequency greater than 100. Large fraction of the dictionary do appear in the source sentences. Hence, we can guarantee that learning to use the dictionary is possible from this dataset.

2.3.2

For the entire corpus and dictionary

LHS freq = 2,90,75,364
LRHS freq = 82,74,281
Constraint copy rate = 1 : 3 (approx)

i.e. for every 3 constraints given by the dictionary 1 constraint is copied into the translation in the training corpus. This is a fair distribution of constraints being copied. Therefore, the dictionary translations are not too common in the data. Model should be able to learn to use the dictionary with this data.

2.3.3

For the words/phrases with fanout = 1

LHS freq = 20,96,848
LRHS freq = 6,27,896
Constraint copy rate = 1 : 3.3 (approx)

For the words/phrases with fanout > 1
LHS freq = 2,69,78,516
LRHS freq = 76,46,385
Constraint copy rate = 1 : 3.5 (approx)

Constraint copying is equally aggressive for both the cases of fanout =1 and > 1, unlike test sets where ratios are even more skewed. This could partly explain the performance issues of fanout =1.
Word-by-word frequencies and analysis can be found here

3 Improving DictDis with Knowledge Distillation

We have kept some initial efforts in improving DictDis using knowledge-distillation methods by creating a student-teacher model. The teacher model would use dictionaries to add the lexical constraints and the student model learns how to disambiguate and select those constraints from the teacher. We wanted to follow the AMAL : Adaptive Mixing of Auxiliary Losses setting.

We have kept it on hold owing to few constraints. If we have a student model for learning disambiguation and copy part, then we have 2 large same-sized models for teacher and student. This makes the training quite time-taking. It also seemed unnecessary as the model itself is deep and copy part is being able learn here itself. Also the process of inference remains in questions as teacher model would be discarded that actually provides the constraints to disambiguate.

4 Lexicon Selection for Lexically Constrained NMT

4.1 Problem Statement

Currently user is asked to select the dictionaries to add lexical constraints from to the translations. We want to make this step automatic. Based on the source text, model should be able to infer the domain and accordingly select the relevant dictionaries to add constraints. Given a set of dictionaries and source text, we aim to select a minimal subset of dictionaries that covers most of the source text.

4.2 Approaches Summary

As a first step, we aim to create a baseline that involves no semantics i.e when we check for coverage of the source by a dictionary, we use simple syntactical token match after lemmatisation instead of matching semantically.

We do not expect constraints of common words to get added. Such words are of least importance and become noise when finding the coverage. To avoid this, we do the following

1. Remove stop words as a pre-processing step after tokenisation
2. Use idfs calculated on Samananthar corpus as weights while finding coverage

4.3 Weighted Overlap

4.3.1 Formulation

A similarity score is calculated for each dictionary. Using a threshold on this similarity score, the top dictionaries are give as the selected ones. This similarity score captures the coverage by calculating number of tokens in the source that occur in the dictionary weighted by idf of the token indicating its uniqueness of appearance.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ corpus}$$

For a phrasal match, we add number of words in phrase instead of single match.

4.3.2 Results

The model is run on several Wikipedia articles and its results (figure 5) are analysed qualitatively. A mixture of dictionaries are being selected - few domain related, general large sized(like chem, defence, eng-hin-gen) along with some noise. But domain-specific dictionaries are not at all picked up. For example, for the article on Dinosaur, zoology is getting selected but palaeontology isn't.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
electrical.csv	electrical.csv	electrical.csv	electrical.csv
chem.csv	nematology.csv	aerospace_glossary.csv	med_comp.csv
aerospace_glossary.csv	aerospace_glossary.csv	geography.csv	aerospace_glossary.csv
geography.csv	geophysics.csv	chem.csv	nematology.csv
nematology.csv	geography.csv	med_comp.csv	geophysics.csv
med_comp.csv	it.csv	it.csv	chem.csv
it.csv	itGlossary.csv	itGlossary.csv	it.csv
itGlossary.csv	med_comp.csv	nematology.csv	itGlossary.csv
geophysics.csv	broadcasting.csv	defence.csv	geography.csv
broadcasting.csv	chem.csv	geophysics.csv	zoology.csv
defence.csv	history.csv	broadcasting.csv	broadcasting.csv
mining_geo.csv	defence.csv	climatology.csv	med.csv
med.csv	itihas_paribhasha.csv	administrative.csv	clinical.csv

Fig. 5: Preference order of dictionaries for Wiki articles given by weighted overlap

4.4 Weighted Overlap with dictionary size normalization

4.4.1 Formulation

For some dictionaries, the size itself can be very large and general, due to which they might achieve undesirably higher similarity score in the previous formulation. To overcome this, a new formulation that performs normalization with dictionary size is used.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ dictionary}$$

4.4.2 Results

The results (figure 6) were very poor. The dictionary size normalisation seems to be overdoing. All the small sized dictionaries were pushed to top leaving no relevant dictionaries in the selection.

4.5 Observations and next steps

We found results of weighted overlap methods unsatisfying. Moreover, it does not model the set cover we desire. Set cover implies subset of dictionaries that together try to cover the entire source. Unlike, weighted overlap

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv
adhoc.csv	geophysics.csv	geophysics.csv	geophysics.csv
geophysics.csv	nematology.csv	nematology.csv	nematology.csv
nematology.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
eng-hindi.csv	adhoc.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi-general.csv	eng-hindi-general.csv	it.csv	it.csv
it.csv	it.csv	itGlossary.csv	itGlossary.csv
itGlossary.csv	itGlossary.csv	broadcasting.csv	broadcasting.csv
broadcasting.csv	itihas_paribhasha.csv	NGMA_Dict.csv	palaeobotany.csv
western_music.csv	broadcasting.csv	maanachitravijnan.csv	western_music.csv
NGMA_Dict.csv	NGMA_Dict.csv	western_music.csv	adhoc.csv
palaeobotany.csv	sociology.csv	kosh_vigyan.csv	NGMA_Dict.csv
agri.csv	western_music.csv	phy.csv	palaeontology.csv
economic_geo.csv	history.csv	physicsGlossary.csv	phy_mal.csv
kosh_vigyan.csv	palaeobotany.csv	administrative.csv	phy.csv

Fig. 6: Preference order of dictionaries for Wiki articles given by weighted overlap with dictionary size normalisation

- Finds amount of overlap of a dictionary independent of the other dictionaries
- Selection of a subset of dictionaries is done based on a given threshold i.e. a minimum amount of overlap required which would give the best dictionaries in terms of "a dictionary alone being able to represent the source"

We hence moved to use the subset selecting sub-modlar functions provided by submidlib. They assign a score for a subset. Based on a given budget i.e. size of subset, the optimal one is found.

4.6 Problem with Graph-cut

For covering a set U, by a subset X of V, Graph-cut function is defined as

$$f_{gc}(X) = \sum_{i \in U, j \in X} s_{ij} - \lambda \sum_{i,j \in X} s_{ij} \tag{4}$$

Matching it to our problem setting

- U is the source text, then i could be modeled as a token
- X is the subset of dictionaries, then j is a dictionary in the subset

Hence, s_{ij} is similarity between a word and an entire dictionary which is not we want to model in our problem. Therefore graph-cut is not the right choice.

4.7 Set Cover

4.7.1 Formulation

It is an implementation of the regular set cover. For covering a set of concepts U, by a subset A of V, its Set Cover evaluation is defined as

$$f(A) = w(\cup_{a \in A} \gamma(a)) = w(\gamma(A)) \tag{5}$$

where $\gamma(A)$ refers to the set of concepts covered by A and $w(\gamma(A))$ is total weight of concepts covered by elements in A

Matching it to our problem setting, the concepts to be covered are tokens in the source, A is the subset of dictionaires, $\gamma(a)$ is the number of tokens of source that occur in dictionary a and w is the idf weighted coverage.

4.7.2 Results

The results (figure 7) are satisfying for a baseline. Domain specific dictionaries are coming in the top. For example maanachitravijnan for Topographic map. General large sized dictionaries(like chem, defence, eng-hin-gen) are coming along with due to their large coverage of the source.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
mining_geo.csv	geography.csv	geography.csv	zoology.csv
chem.csv	electrical.csv	sociology.csv	geography.csv
geography.csv	defence.csv	physicsGlossary.csv	palaeobotany.csv
climatology.csv	med_comp.csv	maanachitravijnan.csv	med_comp.csv
petrology.csv	petrology.csv	chem.csv	chem.csv
sociology.csv	sociology.csv	geology.csv	defence.csv
med_comp.csv	hom_sci.csv	broadcasting.csv	history.csv
geology.csv	itihas_paribhasha.csv	linguistics.csv	bio.csv
bio.csv	electronics.csv	history.csv	botany.csv
mineralogy.csv	lib_info.csv	chem_eng.csv	sociology.csv
chem_eng.csv	clinical.csv	defence.csv	clinical.csv
palaeontology.csv	zoology.csv	psychology.csv	hom_sci.csv
history.csv	quality_cont.csv	med_comp.csv	palaeontology.csv
linguistics.csv	history.csv	bio.csv	electronics.csv

Fig. 7: Preference order of dictionaries for Wiki articles given by Set Cover

Code for all the above implementations can be found here

4.8 Deployment

The SetCover implementation is deployed on the AICTE interfaces (1, 2 and 3)

A challenge in the process of deploying is the position of dictionary selection in the entire pipeline. The selection is to be performed after OCR. Current OCR is highly sophisticated and takes considerable amount of time for large books. Hence providing a preference order of dictionaries and then asking the user to perform selection requires user to intervene in the process after a considerable time.

To overcome this issue, as a current workaround we ask user to give number of dictionaries to auto-select and use them automatically further down the pipeline.

Work is being done on giving user a preference order of dictionaries to select from. This we currently add only for the machine-readable sources on which simple-quick OCR is done to obtain the text.

Code can be found here

5 Future Work

5.1 Extension for multilingual NMT

Current implementation performs pre-processing considering the source language is english. This has to be generalised by performing language-specific pre-processing

Currently subsetting is done from all the available dictionaries. In case of multilingual NMT, there would be dictionaries of multiple source and translation languages. Subsetting should be done only from those dictionaried corresponding to the specified source and translation languages.

5.2 Analysis of current Implementation

Setup profiling feedback on the interface for dictionary selection. Come up with measures that can quantify the results of this functionality.

5.3 Setting up a Learning Problem

Develop the model as a learning problem where we learn the weights of a mixture of multiple submodular and modular functions using max margin as in here.

We seem to need a mixture of different submodular set coverage functions (with normalization, without normalization, with idf, without idf) and several modular functions (quality of each dicitonary, quality of the words being covered - say nouns vs. verbs) and learning mixtures of those

5.4 Keyword based coverage

Translating books is one of the major aims of the pipeline. Books are a structured data, with contents and keywords providing the major information about it. Hence, finding coverage based on just contents/titles/sections/keywords instead of the entire book could work well.

This idea can be incorporated separately to make the subset selection fast. It can also be included in the multiple mixture components in the learning setting.

BTP-I

Ilindra Sai Lakshmi Shreya(190050050)
Guide: Prof. Ganesh Ramakrishnan

Indian Institute of Technology Bombay

1 Study on DictDis

1.1 What is DictDis?

DictDis : Dictionary Constrained Disambiguation for Improved NMT is a lexically constrained NMT system that disambiguates between multiple candidate translations derived from dictionaries. Domain-specific neural machine translation can benefit highly from lexical constraints drawn from domain-specific dictionaries. Prior work has largely focused on the single candidate setting where the target word or phrase is replaced by a single constraint. But dictionaries could present multiple candidate translations for a source word/phrase on account of the polysemous nature of words. DictDis learns to disambiguate the multiple constraints that correspond to a single word/phrase.

The model is trained in a soft manner such that no constraint is forced to appear in the predicted sentence. Given a source sentence and domain specific dictionary constraints consisting of multiple source-target phrases, it can either pick the most relevant constraint or abstain from picking any constraint in order to least affect the fluency of the translation

1.2 Observations

1.2.1 Incomplete Augmented representation

Given a triplet of source sentence, constraints and target sentence (X, C, Y) where $C = \{C_i^j\}$ representing i^{th} inter-phrase constraint and j^{th} intra-phrase constraint, lexical constraints C are added into the pipeline by creating an augmented representation of source sentence and the constraints as follows

$$\hat{X} = [X, \langle sep \rangle, C_1^1, \langle isep \rangle, C_1^2, \langle sep \rangle, C_2^1, ..., C_n, \langle eos \rangle] \tag{1}$$

$\langle sep \rangle$, $\langle isep \rangle$ and $\langle eos \rangle$ are symbols indicating inter-phrase constraint separation, intra-phrase constraint separation and end of sentence respectively.

It leads to the loss of some key information that which source word/phrase does each of the constraints corresponding to. This correspondence could help very much in generating the translations. Therefore, using a representation that can capture this correspondence could help improve the model's performance.

1.2.2 Fanout-based Probability Distribution

Final distribution over target vocabulary is defined as the weighted sum of prediction probability and constraint ingestion probability as follows

$$p(y_t|(y_{<t}, \hat{X})) = g_t P_t^{pred} + (1 - g_t)(P_t^{copy} + P_t^{dis}) \tag{2}$$

P_t^{pred}, P_t^{copy} and P_t^{dis} are prediction probability, copy probability and disambiguate probability respectively.

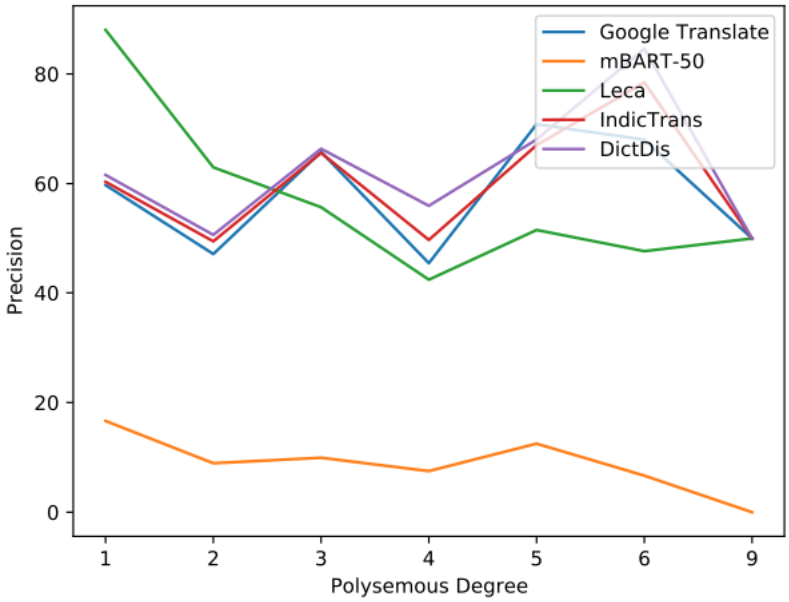

Fig. 1: SPDI for Aerospace dataset

Figures 1, 2, 3 show SPDI for various datasets obtained by various models including DictDis. Clearly the the performance of the model depends on the fanout of the constraints. Modelling fanout-based probability distribution could thererfore help improve the model's performance. One possible way could be adding fanout dependent weights. Currently work is being done in this direction, differentiating the cases of fanout =1 and fanout > 1 as follows

Disamb component :
Case 1 when $f_{t=1} : (1 - g_t)w(p_t^{copy} + p_t^{dis})$
Case 2 when $f_{t>1} : (1 - g_t)(1 - w)(p_t^{copy} + p_t^{dis})$

$$p(y_t|y_{<t}, X) = Disamb\ Component + g_t p_t^{predict} \tag{3}$$

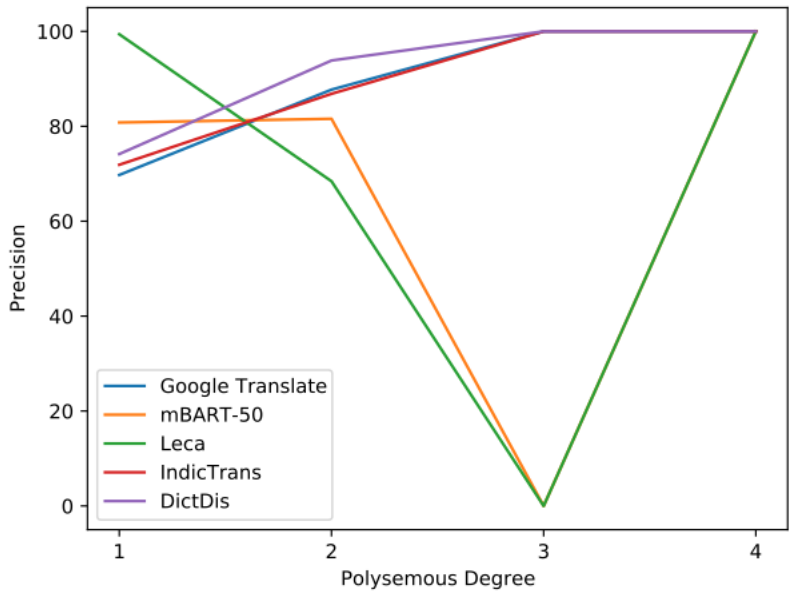

Fig. 2: SPDI for Banking dataset

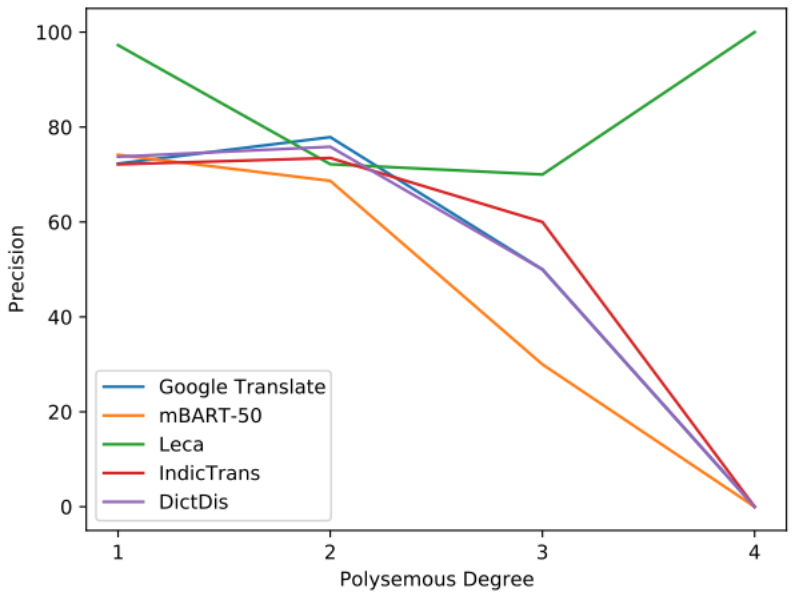

Fig. 3: SPDI for Regulatory dataset

1.2.3 Dictionary Usage by the Model

SPDI(figures 1, 2, 3) for lower fanout, especially at 1 in any domain is lower for DictDis compared to Leca. It is at the level of IndicTrans, a model which does not use any dictionary. Bleu(figure 4) scores for Banking and Regulatory domains are better for other models. These are the domains with less ambiguity i.e most of the constraints would have lower fanout.

	Regulatory	Aerospace	Banking	Combustion-T1	Combustion-T2	Structures	Medical
GoogleTrans	24.9	36	31.3	43.2	40.1	43	15.6
mBART	24.7	28.1	28.1	31.1	28.9	34.2	17
Leca	**28.4**	26.9	32.6	30.4	38.9	37.9	16.3
IndicTrans	27.3	37.3	**34.4**	43.3	41.3	47.2	15.7
DictDis	27.5	**37.9**	34	**44.9**	**43.4**	**51.3**	**20.6**

Fig. 4: BLEU scores for various datasets and approaches

These observations bring in the following questions

1. Whether dictionary is being used at all by the model?
   - Are the translations present in dictionary very common in the dataset that the model picks correct translation from the data itself rather than picking from dictionary?
2. Why the model is performing comparatively low when fanout =1?
   - For fanout 1 replacement(vs. no replacement) seems more aggressive on test sets relative to the replacement of fanout > 1, does it remain same for training sets?

To understand and answer these questions, we performed frequency analysis on the occurrence of dictionary words/phrases in the corpus.

2 Frequency Analysis of Dictionary Data in Samananthar Dataset

2.1 Problem Statement

Pondering the question why when fanout is 1, DictDis remained at the level of IndicTrans which does not use any dictionary. We want to check whether the model is getting to learn to use the dictionary and pick constraints from it at all. We check the frequency of occurrences of dictionary words/phrases in the training corpus. This would show where the problem exactly is

- Distribution and diversity of the dictionary dataset and training corpus used
- (or) The model itself

2.2 Processing

Following quantities are calculated (code) on and the training corpus, Samananthar and the generalised en-hi dictionary used for training

1. No. of LHS words/phrases of dictionary with non-zero occurrences in corpus
2. Frequency of LHS words/phrases in the source sentences (call it "LHS freq")
3. Frequency of LHS words/phrases in the source sentences when RHS translations occur in corresponding target sentences (call it "LRHS freq")

2.3 Results and Observations

2.3.1

Number of words/phrases in the dictionary = 11,767
Among these 11,593 have non-zero occurrences and 7,799 i.e approximately 75% have frequency greater than 100. Large fraction of the dictionary do appear in the source sentences. Hence, we can guarantee that learning to use the dictionary is possible from this dataset.

2.3.2

For the entire corpus and dictionary

LHS freq = 2,90,75,364
LRHS freq = 82,74,281
Constraint copy rate = 1 : 3 (approx)

i.e. for every 3 constraints given by the dictionary 1 constraint is copied into the translation in the training corpus. This is a fair distribution of constraints being copied. Therefore, the dictionary translations are not too common in the data. Model should be able to learn to use the dictionary with this data.

2.3.3

For the words/phrases with fanout = 1

LHS freq = 20,96,848
LRHS freq = 6,27,896
Constraint copy rate = 1 : 3.3 (approx)

For the words/phrases with fanout > 1
LHS freq = 2,69,78,516
LRHS freq = 76,46,385
Constraint copy rate = 1 : 3.5 (approx)

Constraint copying is equally aggressive for both the cases of fanout =1 and > 1, unlike test sets where ratios are even more skewed. This could partly explain the performance issues of fanout =1.
Word-by-word frequencies and analysis can be found here

3 Improving DictDis with Knowledge Distillation

We have kept some initial efforts in improving DictDis using knowledge-distillation methods by creating a student-teacher model. The teacher model would use dictionaries to add the lexical constraints and the student model learns how to disambiguate and select those constraints from the teacher. We wanted to follow the AMAL : Adaptive Mixing of Auxiliary Losses setting.

We have kept it on hold owing to few constraints. If we have a student model for learning disambiguation and copy part, then we have 2 large same-sized models for teacher and student. This makes the training quite time-taking. It also seemed unnecessary as the model itself is deep and copy part is being able learn here itself. Also the process of inference remains in questions as teacher model would be discarded that actually provides the constraints to disambiguate.

4 Lexicon Selection for Lexically Constrained NMT

4.1 Problem Statement

Currently user is asked to select the dictionaries to add lexical constraints from to the translations. We want to make this step automatic. Based on the source text, model should be able to infer the domain and accordingly select the relevant dictionaries to add constraints. Given a set of dictionaries and source text, we aim to select a minimal subset of dictionaries that covers most of the source text.

4.2 Approaches Summary

As a first step, we aim to create a baseline that involves no semantics i.e when we check for coverage of the source by a dictionary, we use simple syntactical token match after lemmatisation instead of matching semantically.

We do not expect constraints of common words to get added. Such words are of least importance and become noise when finding the coverage. To avoid this, we do the following

1. Remove stop words as a pre-processing step after tokenisation
2. Use idfs calculated on Samananthar corpus as weights while finding coverage

4.3 Weighted Overlap

4.3.1 Formulation

A similarity score is calculated for each dictionary. Using a threshold on this similarity score, the top dictionaries are give as the selected ones. This similarity score captures the coverage by calculating number of tokens in the source that occur in the dictionary weighted by idf of the token indicating its uniqueness of appearance.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ corpus}$$

For a phrasal match, we add number of words in phrase instead of single match.

4.3.2 Results

The model is run on several Wikipedia articles and its results (figure 5) are analysed qualitatively. A mixture of dictionaries are being selected - few domain related, general large sized(like chem, defence, eng-hin-gen) along with some noise. But domain-specific dictionaries are not at all picked up. For example, for the article on Dinosaur, zoology is getting selected but palaeontology isn't.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
electrical.csv	electrical.csv	electrical.csv	electrical.csv
chem.csv	nematology.csv	aerospace_glossary.csv	med_comp.csv
aerospace_glossary.csv	aerospace_glossary.csv	geography.csv	aerospace_glossary.csv
geography.csv	geophysics.csv	chem.csv	nematology.csv
nematology.csv	geography.csv	med_comp.csv	geophysics.csv
med_comp.csv	it.csv	it.csv	chem.csv
it.csv	itGlossary.csv	itGlossary.csv	it.csv
itGlossary.csv	med_comp.csv	nematology.csv	itGlossary.csv
geophysics.csv	broadcasting.csv	defence.csv	geography.csv
broadcasting.csv	chem.csv	geophysics.csv	zoology.csv
defence.csv	history.csv	broadcasting.csv	broadcasting.csv
mining_geo.csv	defence.csv	climatology.csv	med.csv
med.csv	itihas_paribhasha.csv	administrative.csv	clinical.csv

Fig. 5: Preference order of dictionaries for Wiki articles given by weighted overlap

4.4 Weighted Overlap with dictionary size normalization

4.4.1 Formulation

For some dictionaries, the size itself can be very large and general, due to which they might achieve undesirably higher similarity score in the previous formulation. To overcome this, a new formulation that performs normalization with dictionary size is used.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ dictionary}$$

4.4.2 Results

The results (figure 6) were very poor. The dictionary size normalisation seems to be overdoing. All the small sized dictionaries were pushed to top leaving no relevant dictionaries in the selection.

4.5 Observations and next steps

We found results of weighted overlap methods unsatisfying. Moreover, it does not model the set cover we desire. Set cover implies subset of dictionaries that together try to cover the entire source. Unlike, weighted overlap

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv
adhoc.csv	geophysics.csv	geophysics.csv	geophysics.csv
geophysics.csv	nematology.csv	nematology.csv	nematology.csv
nematology.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
eng-hindi.csv	adhoc.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi-general.csv	eng-hindi-general.csv	it.csv	it.csv
it.csv	it.csv	itGlossary.csv	itGlossary.csv
itGlossary.csv	itGlossary.csv	broadcasting.csv	broadcasting.csv
broadcasting.csv	itihas_paribhasha.csv	NGMA_Dict.csv	palaeobotany.csv
western_music.csv	broadcasting.csv	maanachitravijnan.csv	western_music.csv
NGMA_Dict.csv	NGMA_Dict.csv	western_music.csv	adhoc.csv
palaeobotany.csv	sociology.csv	kosh_vigyan.csv	NGMA_Dict.csv
agri.csv	western_music.csv	phy.csv	palaeontology.csv
economic_geo.csv	history.csv	physicsGlossary.csv	phy_mal.csv
kosh_vigyan.csv	palaeobotany.csv	administrative.csv	phy.csv

Fig. 6: Preference order of dictionaries for Wiki articles given by weighted overlap with dictionary size normalisation

- Finds amount of overlap of a dictionary independent of the other dictionaries
- Selection of a subset of dictionaries is done based on a given threshold i.e. a minimum amount of overlap required which would give the best dictionaries in terms of "a dictionary alone being able to represent the source"

We hence moved to use the subset selecting sub-modlar functions provided by submidlib. They assign a score for a subset. Based on a given budget i.e. size of subset, the optimal one is found.

4.6 Problem with Graph-cut

For covering a set U, by a subset X of V, Graph-cut function is defined as

$$f_{gc}(X) = \sum_{i \in U, j \in X} s_{ij} - \lambda \sum_{i,j \in X} s_{ij} \tag{4}$$

Matching it to our problem setting

- U is the source text, then i could be modeled as a token
- X is the subset of dictionaries, then j is a dictionary in the subset

Hence, s_{ij} is similarity between a word and an entire dictionary which is not we want to model in our problem. Therefore graph-cut is not the right choice.

4.7 Set Cover

4.7.1 Formulation

It is an implementation of the regular set cover. For covering a set of concepts U, by a subset A of V, its Set Cover evaluation is defined as

$$f(A) = w(\cup_{a \in A} \gamma(a)) = w(\gamma(A)) \tag{5}$$

where $\gamma(A)$ refers to the set of concepts covered by A and $w(\gamma(A))$ is total weight of concepts covered by elements in A

Matching it to our problem setting, the concepts to be covered are tokens in the source, A is the subset of dictionaires, $\gamma(a)$ is the number of tokens of source that occur in dictionary a and w is the idf weighted coverage.

4.7.2 Results

The results (figure 7) are satisfying for a baseline. Domain specific dictionaries are coming in the top. For example maanachitravijnan for Topographic map. General large sized dictionaries(like chem, defence, eng-hin-gen) are coming along with due to their large coverage of the source.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
mining_geo.csv	geography.csv	geography.csv	zoology.csv
chem.csv	electrical.csv	sociology.csv	geography.csv
geography.csv	defence.csv	physicsGlossary.csv	palaeobotany.csv
climatology.csv	med_comp.csv	maanachitravijnan.csv	med_comp.csv
petrology.csv	petrology.csv	chem.csv	chem.csv
sociology.csv	sociology.csv	geology.csv	defence.csv
med_comp.csv	hom_sci.csv	broadcasting.csv	history.csv
geology.csv	itihas_paribhasha.csv	linguistics.csv	bio.csv
bio.csv	electronics.csv	history.csv	botany.csv
mineralogy.csv	lib_info.csv	chem_eng.csv	sociology.csv
chem_eng.csv	clinical.csv	defence.csv	clinical.csv
palaeontology.csv	zoology.csv	psychology.csv	hom_sci.csv
history.csv	quality_cont.csv	med_comp.csv	palaeontology.csv
linguistics.csv	history.csv	bio.csv	electronics.csv

Fig. 7: Preference order of dictionaries for Wiki articles given by Set Cover

Code for all the above implementations can be found here

4.8 Deployment

The SetCover implementation is deployed on the AICTE interfaces (1, 2 and 3)

A challenge in the process of deploying is the position of dictionary selection in the entire pipeline. The selection is to be performed after OCR. Current OCR is highly sophisticated and takes considerable amount of time for large books. Hence providing a preference order of dictionaries and then asking the user to perform selection requires user to intervene in the process after a considerable time.

To overcome this issue, as a current workaround we ask user to give number of dictionaries to auto-select and use them automatically further down the pipeline.

Work is being done on giving user a preference order of dictionaries to select from. This we currently add only for the machine-readable sources on which simple-quick OCR is done to obtain the text.

Code can be found here

5 Future Work

5.1 Extension for multilingual NMT

Current implementation performs pre-processing considering the source language is english. This has to be generalised by performing language-specific pre-processing

Currently subsetting is done from all the available dictionaries. In case of multilingual NMT, there would be dictionaries of multiple source and translation languages. Subsetting should be done only from those dictionaried corresponding to the specified source and translation languages.

5.2 Analysis of current Implementation

Setup profiling feedback on the interface for dictionary selection. Come up with measures that can quantify the results of this functionality.

5.3 Setting up a Learning Problem

Develop the model as a learning problem where we learn the weights of a mixture of multiple submodular and modular functions using max margin as in here.

We seem to need a mixture of different submodular set coverage functions (with normalization, without normalization, with idf, without idf) and several modular functions (quality of each dicitonary, quality of the words being covered - say nouns vs. verbs) and learning mixtures of those

5.4 Keyword based coverage

Translating books is one of the major aims of the pipeline. Books are a structured data, with contents and keywords providing the major information about it. Hence, finding coverage based on just contents/titles/sections/keywords instead of the entire book could work well.

This idea can be incorporated separately to make the subset selection fast. It can also be included in the multiple mixture components in the learning setting.

BTP-I

Ilindra Sai Lakshmi Shreya(190050050)
Guide: Prof. Ganesh Ramakrishnan

Indian Institute of Technology Bombay

1 Study on DictDis

1.1 What is DictDis?

DictDis : Dictionary Constrained Disambiguation for Improved NMT is a lexically constrained NMT system that disambiguates between multiple candidate translations derived from dictionaries. Domain-specific neural machine translation can benefit highly from lexical constraints drawn from domain-specific dictionaries. Prior work has largely focused on the single candidate setting where the target word or phrase is replaced by a single constraint. But dictionaries could present multiple candidate translations for a source word/phrase on account of the polysemous nature of words. DictDis learns to disambiguate the multiple constraints that correspond to a single word/phrase.

The model is trained in a soft manner such that no constraint is forced to appear in the predicted sentence. Given a source sentence and domain specific dictionary constraints consisting of multiple source-target phrases, it can either pick the most relevant constraint or abstain from picking any constraint in order to least affect the fluency of the translation

1.2 Observations

1.2.1 Incomplete Augmented representation

Given a triplet of source sentence, constraints and target sentence (X, C, Y) where $C = \{C_i^j\}$ representing i^{th} inter-phrase constraint and j^{th} intra-phrase constraint, lexical constraints C are added into the pipeline by creating an augmented representation of source sentence and the constraints as follows

$$\hat{X} = [X, \langle sep \rangle, C_1^1, \langle isep \rangle, C_1^2, \langle sep \rangle, C_2^1, ..., C_n, \langle eos \rangle] \tag{1}$$

$\langle sep \rangle$, $\langle isep \rangle$ and $\langle eos \rangle$ are symbols indicating inter-phrase constraint separation, intra-phrase constraint separation and end of sentence respectively.

It leads to the loss of some key information that which source word/phrase does each of the constraints corresponding to. This correspondence could help very much in generating the translations. Therefore, using a representation that can capture this correspondence could help improve the model's performance.

1.2.2 Fanout-based Probability Distribution

Final distribution over target vocabulary is defined as the weighted sum of prediction probability and constraint ingestion probability as follows

$$p(y_t|(y_{<t}, \hat{X})) = g_t P_t^{pred} + (1 - g_t)(P_t^{copy} + P_t^{dis}) \tag{2}$$

P_t^{pred}, P_t^{copy} and P_t^{dis} are prediction probability, copy probability and disambiguate probability respectively.

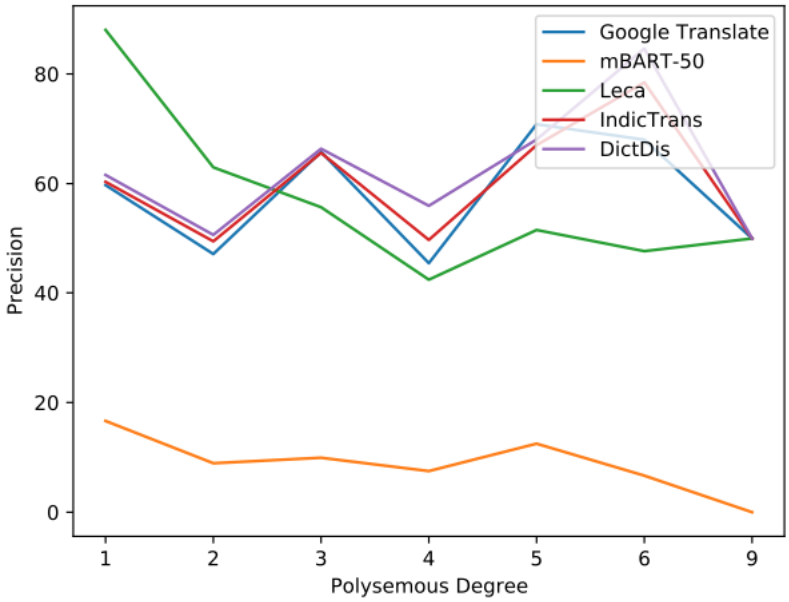

Fig. 1: SPDI for Aerospace dataset

Figures 1, 2, 3 show SPDI for various datasets obtained by various models including DictDis. Clearly the the performance of the model depends on the fanout of the constraints. Modelling fanout-based probability distribution could thererfore help improve the model's performance. One possible way could be adding fanout dependent weights. Currently work is being done in this direction, differentiating the cases of fanout =1 and fanout > 1 as follows

Disamb component :
Case 1 when $f_{t=1} : (1 - g_t)w(p_t^{copy} + p_t^{dis})$
Case 2 when $f_{t>1} : (1 - g_t)(1 - w)(p_t^{copy} + p_t^{dis})$

$$p(y_t|y_{<t}, X) = Disamb\ Component + g_t p_t^{predict} \tag{3}$$

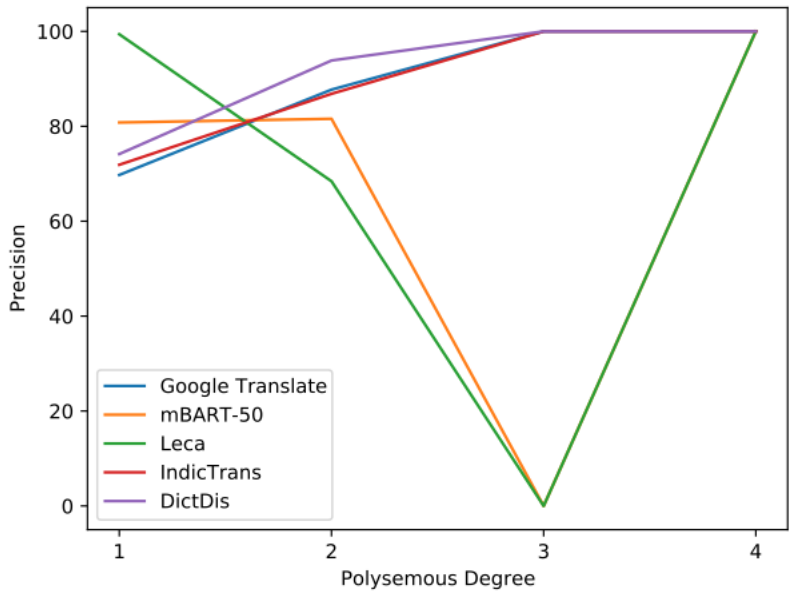

Fig. 2: SPDI for Banking dataset

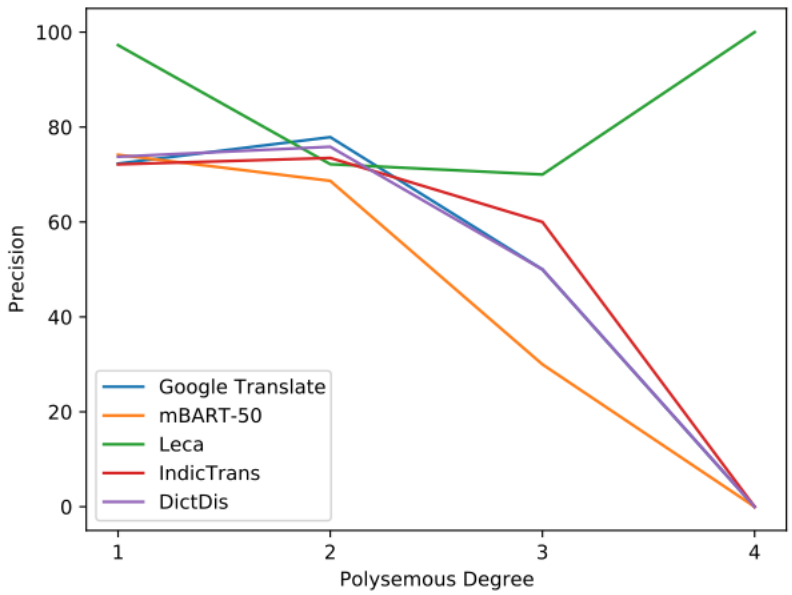

Fig. 3: SPDI for Regulatory dataset

1.2.3 Dictionary Usage by the Model

SPDI(figures 1, 2, 3) for lower fanout, especially at 1 in any domain is lower for DictDis compared to Leca. It is at the level of IndicTrans, a model which does not use any dictionary. Bleu(figure 4) scores for Banking and Regulatory domains are better for other models. These are the domains with less ambiguity i.e most of the constraints would have lower fanout.

	Regulatory	Aerospace	Banking	Combustion-T1	Combustion-T2	Structures	Medical
GoogleTrans	24.9	36	31.3	43.2	40.1	43	15.6
mBART	24.7	28.1	28.1	31.1	28.9	34.2	17
Leca	**28.4**	26.9	32.6	30.4	38.9	37.9	16.3
IndicTrans	27.3	37.3	**34.4**	43.3	41.3	47.2	15.7
DictDis	27.5	**37.9**	34	**44.9**	**43.4**	**51.3**	**20.6**

Fig. 4: BLEU scores for various datasets and approaches

These observations bring in the following questions

1. Whether dictionary is being used at all by the model?
   - Are the translations present in dictionary very common in the dataset that the model picks correct translation from the data itself rather than picking from dictionary?
2. Why the model is performing comparatively low when fanout =1?
   - For fanout 1 replacement(vs. no replacement) seems more aggressive on test sets relative to the replacement of fanout > 1, does it remain same for training sets?

To understand and answer these questions, we performed frequency analysis on the occurrence of dictionary words/phrases in the corpus.

2 Frequency Analysis of Dictionary Data in Samananthar Dataset

2.1 Problem Statement

Pondering the question why when fanout is 1, DictDis remained at the level of IndicTrans which does not use any dictionary. We want to check whether the model is getting to learn to use the dictionary and pick constraints from it at all. We check the frequency of occurrences of dictionary words/phrases in the training corpus. This would show where the problem exactly is

- Distribution and diversity of the dictionary dataset and training corpus used
- (or) The model itself

2.2 Processing

Following quantities are calculated (code) on and the training corpus, Samananthar and the generalised en-hi dictionary used for training

1. No. of LHS words/phrases of dictionary with non-zero occurrences in corpus
2. Frequency of LHS words/phrases in the source sentences (call it "LHS freq")
3. Frequency of LHS words/phrases in the source sentences when RHS translations occur in corresponding target sentences (call it "LRHS freq")

2.3 Results and Observations

2.3.1

Number of words/phrases in the dictionary = 11,767
Among these 11,593 have non-zero occurrences and 7,799 i.e approximately 75% have frequency greater than 100. Large fraction of the dictionary do appear in the source sentences. Hence, we can guarantee that learning to use the dictionary is possible from this dataset.

2.3.2

For the entire corpus and dictionary

LHS freq = 2,90,75,364
LRHS freq = 82,74,281
Constraint copy rate = 1 : 3 (approx)

i.e. for every 3 constraints given by the dictionary 1 constraint is copied into the translation in the training corpus. This is a fair distribution of constraints being copied. Therefore, the dictionary translations are not too common in the data. Model should be able to learn to use the dictionary with this data.

2.3.3

For the words/phrases with fanout = 1

LHS freq = 20,96,848
LRHS freq = 6,27,896
Constraint copy rate = 1 : 3.3 (approx)

For the words/phrases with fanout > 1
LHS freq = 2,69,78,516
LRHS freq = 76,46,385
Constraint copy rate = 1 : 3.5 (approx)

Constraint copying is equally aggressive for both the cases of fanout =1 and > 1, unlike test sets where ratios are even more skewed. This could partly explain the performance issues of fanout =1.
Word-by-word frequencies and analysis can be found here

3 Improving DictDis with Knowledge Distillation

We have kept some initial efforts in improving DictDis using knowledge-distillation methods by creating a student-teacher model. The teacher model would use dictionaries to add the lexical constraints and the student model learns how to disambiguate and select those constraints from the teacher. We wanted to follow the AMAL : Adaptive Mixing of Auxiliary Losses setting.

We have kept it on hold owing to few constraints. If we have a student model for learning disambiguation and copy part, then we have 2 large same-sized models for teacher and student. This makes the training quite time-taking. It also seemed unnecessary as the model itself is deep and copy part is being able learn here itself. Also the process of inference remains in questions as teacher model would be discarded that actually provides the constraints to disambiguate.

4 Lexicon Selection for Lexically Constrained NMT

4.1 Problem Statement

Currently user is asked to select the dictionaries to add lexical constraints from to the translations. We want to make this step automatic. Based on the source text, model should be able to infer the domain and accordingly select the relevant dictionaries to add constraints. Given a set of dictionaries and source text, we aim to select a minimal subset of dictionaries that covers most of the source text.

4.2 Approaches Summary

As a first step, we aim to create a baseline that involves no semantics i.e when we check for coverage of the source by a dictionary, we use simple syntactical token match after lemmatisation instead of matching semantically.

We do not expect constraints of common words to get added. Such words are of least importance and become noise when finding the coverage. To avoid this, we do the following

1. Remove stop words as a pre-processing step after tokenisation
2. Use idfs calculated on Samananthar corpus as weights while finding coverage

4.3 Weighted Overlap

4.3.1 Formulation

A similarity score is calculated for each dictionary. Using a threshold on this similarity score, the top dictionaries are give as the selected ones. This similarity score captures the coverage by calculating number of tokens in the source that occur in the dictionary weighted by idf of the token indicating its uniqueness of appearance.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ corpus}$$

For a phrasal match, we add number of words in phrase instead of single match.

4.3.2 Results

The model is run on several Wikipedia articles and its results (figure 5) are analysed qualitatively. A mixture of dictionaries are being selected - few domain related, general large sized(like chem, defence, eng-hin-gen) along with some noise. But domain-specific dictionaries are not at all picked up. For example, for the article on Dinosaur, zoology is getting selected but palaeontology isn't.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
electrical.csv	electrical.csv	electrical.csv	electrical.csv
chem.csv	nematology.csv	aerospace_glossary.csv	med_comp.csv
aerospace_glossary.csv	aerospace_glossary.csv	geography.csv	aerospace_glossary.csv
geography.csv	geophysics.csv	chem.csv	nematology.csv
nematology.csv	geography.csv	med_comp.csv	geophysics.csv
med_comp.csv	it.csv	it.csv	chem.csv
it.csv	itGlossary.csv	itGlossary.csv	it.csv
itGlossary.csv	med_comp.csv	nematology.csv	itGlossary.csv
geophysics.csv	broadcasting.csv	defence.csv	geography.csv
broadcasting.csv	chem.csv	geophysics.csv	zoology.csv
defence.csv	history.csv	broadcasting.csv	broadcasting.csv
mining_geo.csv	defence.csv	climatology.csv	med.csv
med.csv	itihas_paribhasha.csv	administrative.csv	clinical.csv

Fig. 5: Preference order of dictionaries for Wiki articles given by weighted overlap

4.4 Weighted Overlap with dictionary size normalization

4.4.1 Formulation

For some dictionaries, the size itself can be very large and general, due to which they might achieve undesirably higher similarity score in the previous formulation. To overcome this, a new formulation that performs normalization with dictionary size is used.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ dictionary}$$

4.4.2 Results

The results (figure 6) were very poor. The dictionary size normalisation seems to be overdoing. All the small sized dictionaries were pushed to top leaving no relevant dictionaries in the selection.

4.5 Observations and next steps

We found results of weighted overlap methods unsatisfying. Moreover, it does not model the set cover we desire. Set cover implies subset of dictionaries that together try to cover the entire source. Unlike, weighted overlap

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv
adhoc.csv	geophysics.csv	geophysics.csv	geophysics.csv
geophysics.csv	nematology.csv	nematology.csv	nematology.csv
nematology.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
eng-hindi.csv	adhoc.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi-general.csv	eng-hindi-general.csv	it.csv	it.csv
it.csv	it.csv	itGlossary.csv	itGlossary.csv
itGlossary.csv	itGlossary.csv	broadcasting.csv	broadcasting.csv
broadcasting.csv	itihas_paribhasha.csv	NGMA_Dict.csv	palaeobotany.csv
western_music.csv	broadcasting.csv	maanachitravijnan.csv	western_music.csv
NGMA_Dict.csv	NGMA_Dict.csv	western_music.csv	adhoc.csv
palaeobotany.csv	sociology.csv	kosh_vigyan.csv	NGMA_Dict.csv
agri.csv	western_music.csv	phy.csv	palaeontology.csv
economic_geo.csv	history.csv	physicsGlossary.csv	phy_mal.csv
kosh_vigyan.csv	palaeobotany.csv	administrative.csv	phy.csv

Fig. 6: Preference order of dictionaries for Wiki articles given by weighted overlap with dictionary size normalisation

- Finds amount of overlap of a dictionary independent of the other dictionaries
- Selection of a subset of dictionaries is done based on a given threshold i.e. a minimum amount of overlap required which would give the best dictionaries in terms of "a dictionary alone being able to represent the source"

We hence moved to use the subset selecting sub-modlar functions provided by submidlib. They assign a score for a subset. Based on a given budget i.e. size of subset, the optimal one is found.

4.6 Problem with Graph-cut

For covering a set U, by a subset X of V, Graph-cut function is defined as

$$f_{gc}(X) = \sum_{i \in U, j \in X} s_{ij} - \lambda \sum_{i,j \in X} s_{ij} \tag{4}$$

Matching it to our problem setting

- U is the source text, then i could be modeled as a token
- X is the subset of dictionaries, then j is a dictionary in the subset

Hence, s_{ij} is similarity between a word and an entire dictionary which is not we want to model in our problem. Therefore graph-cut is not the right choice.

4.7 Set Cover

4.7.1 Formulation

It is an implementation of the regular set cover. For covering a set of concepts U, by a subset A of V, its Set Cover evaluation is defined as

$$f(A) = w(\cup_{a \in A} \gamma(a)) = w(\gamma(A)) \tag{5}$$

where $\gamma(A)$ refers to the set of concepts covered by A and $w(\gamma(A))$ is total weight of concepts covered by elements in A

Matching it to our problem setting, the concepts to be covered are tokens in the source, A is the subset of dictionaires, $\gamma(a)$ is the number of tokens of source that occur in dictionary a and w is the idf weighted coverage.

4.7.2 Results

The results (figure 7) are satisfying for a baseline. Domain specific dictionaries are coming in the top. For example maanachitravijnan for Topographic map. General large sized dictionaries(like chem, defence, eng-hin-gen) are coming along with due to their large coverage of the source.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
mining_geo.csv	geography.csv	geography.csv	zoology.csv
chem.csv	electrical.csv	sociology.csv	geography.csv
geography.csv	defence.csv	physicsGlossary.csv	palaeobotany.csv
climatology.csv	med_comp.csv	maanachitravijnan.csv	med_comp.csv
petrology.csv	petrology.csv	chem.csv	chem.csv
sociology.csv	sociology.csv	geology.csv	defence.csv
med_comp.csv	hom_sci.csv	broadcasting.csv	history.csv
geology.csv	itihas_paribhasha.csv	linguistics.csv	bio.csv
bio.csv	electronics.csv	history.csv	botany.csv
mineralogy.csv	lib_info.csv	chem_eng.csv	sociology.csv
chem_eng.csv	clinical.csv	defence.csv	clinical.csv
palaeontology.csv	zoology.csv	psychology.csv	hom_sci.csv
history.csv	quality_cont.csv	med_comp.csv	palaeontology.csv
linguistics.csv	history.csv	bio.csv	electronics.csv

Fig. 7: Preference order of dictionaries for Wiki articles given by Set Cover

Code for all the above implementations can be found here

4.8 Deployment

The SetCover implementation is deployed on the AICTE interfaces (1, 2 and 3)

A challenge in the process of deploying is the position of dictionary selection in the entire pipeline. The selection is to be performed after OCR. Current OCR is highly sophisticated and takes considerable amount of time for large books. Hence providing a preference order of dictionaries and then asking the user to perform selection requires user to intervene in the process after a considerable time.

To overcome this issue, as a current workaround we ask user to give number of dictionaries to auto-select and use them automatically further down the pipeline.

Work is being done on giving user a preference order of dictionaries to select from. This we currently add only for the machine-readable sources on which simple-quick OCR is done to obtain the text.

Code can be found here

5 Future Work

5.1 Extension for multilingual NMT

Current implementation performs pre-processing considering the source language is english. This has to be generalised by performing language-specific pre-processing

Currently subsetting is done from all the available dictionaries. In case of multilingual NMT, there would be dictionaries of multiple source and translation languages. Subsetting should be done only from those dictionaried corresponding to the specified source and translation languages.

5.2 Analysis of current Implementation

Setup profiling feedback on the interface for dictionary selection. Come up with measures that can quantify the results of this functionality.

5.3 Setting up a Learning Problem

Develop the model as a learning problem where we learn the weights of a mixture of multiple submodular and modular functions using max margin as in here.

We seem to need a mixture of different submodular set coverage functions (with normalization, without normalization, with idf, without idf) and several modular functions (quality of each dicitonary, quality of the words being covered - say nouns vs. verbs) and learning mixtures of those

5.4 Keyword based coverage

Translating books is one of the major aims of the pipeline. Books are a structured data, with contents and keywords providing the major information about it. Hence, finding coverage based on just contents/titles/sections/keywords instead of the entire book could work well.

This idea can be incorporated separately to make the subset selection fast. It can also be included in the multiple mixture components in the learning setting.

BTP-I

Ilindra Sai Lakshmi Shreya(190050050)
Guide: Prof. Ganesh Ramakrishnan

Indian Institute of Technology Bombay

1 Study on DictDis

1.1 What is DictDis?

DictDis : Dictionary Constrained Disambiguation for Improved NMT is a lexically constrained NMT system that disambiguates between multiple candidate translations derived from dictionaries. Domain-specific neural machine translation can benefit highly from lexical constraints drawn from domain-specific dictionaries. Prior work has largely focused on the single candidate setting where the target word or phrase is replaced by a single constraint. But dictionaries could present multiple candidate translations for a source word/phrase on account of the polysemous nature of words. DictDis learns to disambiguate the multiple constraints that correspond to a single word/phrase.

The model is trained in a soft manner such that no constraint is forced to appear in the predicted sentence. Given a source sentence and domain specific dictionary constraints consisting of multiple source-target phrases, it can either pick the most relevant constraint or abstain from picking any constraint in order to least affect the fluency of the translation

1.2 Observations

1.2.1 Incomplete Augmented representation

Given a triplet of source sentence, constraints and target sentence (X, C, Y) where $C = \{C_i^j\}$ representing i^{th} inter-phrase constraint and j^{th} intra-phrase constraint, lexical constraints C are added into the pipeline by creating an augmented representation of source sentence and the constraints as follows

$$\hat{X} = [X, \langle sep \rangle, C_1^1, \langle isep \rangle, C_1^2, \langle sep \rangle, C_2^1, ..., C_n, \langle eos \rangle] \tag{1}$$

$\langle sep \rangle$, $\langle isep \rangle$ and $\langle eos \rangle$ are symbols indicating inter-phrase constraint separation, intra-phrase constraint separation and end of sentence respectively.

It leads to the loss of some key information that which source word/phrase does each of the constraints corresponding to. This correspondence could help very much in generating the translations. Therefore, using a representation that can capture this correspondence could help improve the model's performance.

1.2.2 Fanout-based Probability Distribution

Final distribution over target vocabulary is defined as the weighted sum of prediction probability and constraint ingestion probability as follows

$$p(y_t|(y_{<t}, \hat{X})) = g_t P_t^{pred} + (1 - g_t)(P_t^{copy} + P_t^{dis}) \tag{2}$$

P_t^{pred}, P_t^{copy} and P_t^{dis} are prediction probability, copy probability and disambiguate probability respectively.

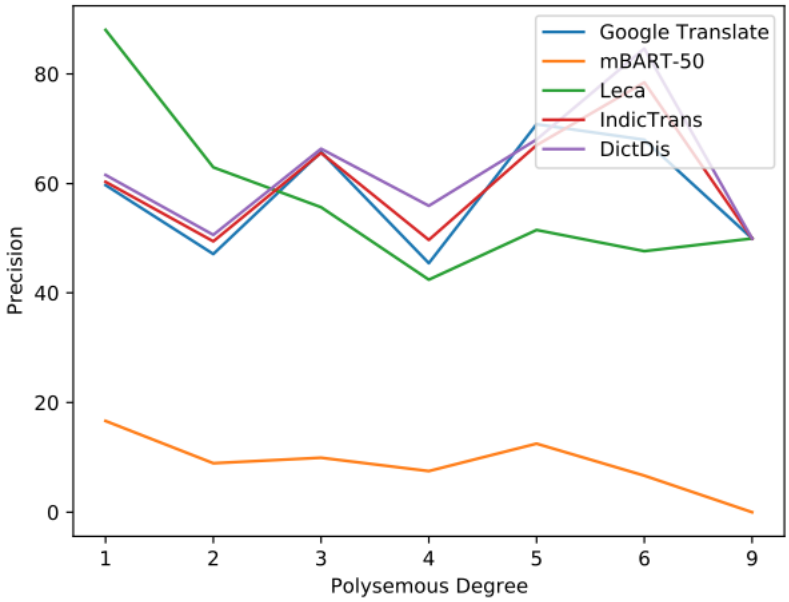

Fig. 1: SPDI for Aerospace dataset

Figures 1, 2, 3 show SPDI for various datasets obtained by various models including DictDis. Clearly the the performance of the model depends on the fanout of the constraints. Modelling fanout-based probability distribution could thererfore help improve the model's performance. One possible way could be adding fanout dependent weights. Currently work is being done in this direction, differentiating the cases of fanout =1 and fanout > 1 as follows

Disamb component :
Case 1 when $f_{t=1} : (1 - g_t)w(p_t^{copy} + p_t^{dis})$
Case 2 when $f_{t>1} : (1 - g_t)(1 - w)(p_t^{copy} + p_t^{dis})$

$$p(y_t|y_{<t}, X) = Disamb\ Component + g_t p_t^{predict} \tag{3}$$

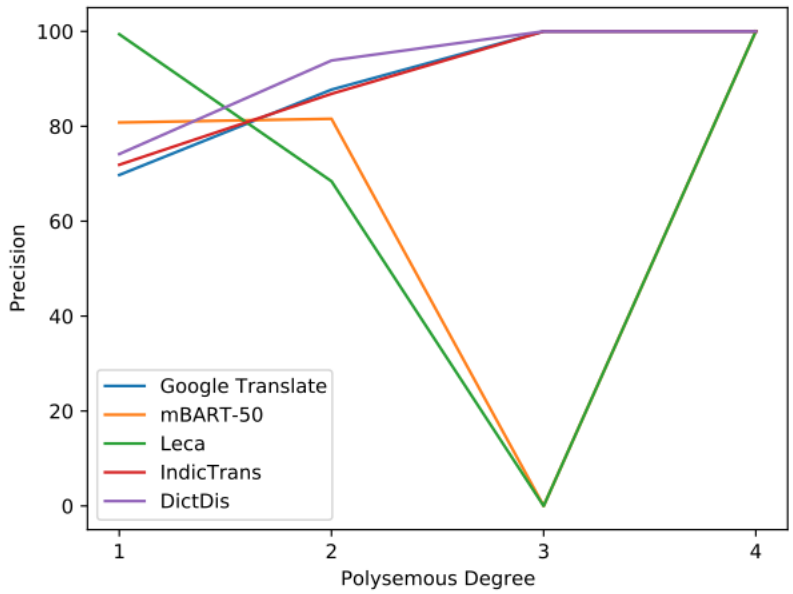

Fig. 2: SPDI for Banking dataset

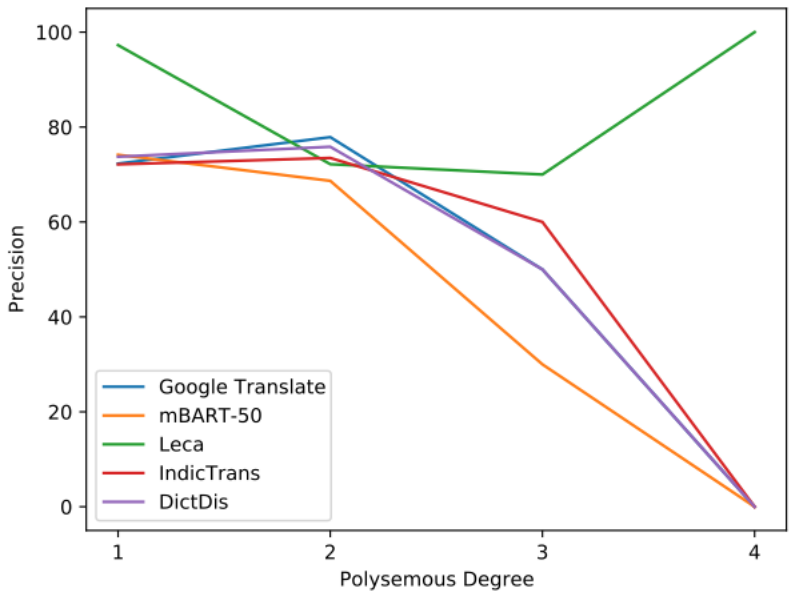

Fig. 3: SPDI for Regulatory dataset

1.2.3 Dictionary Usage by the Model

SPDI(figures 1, 2, 3) for lower fanout, especially at 1 in any domain is lower for DictDis compared to Leca. It is at the level of IndicTrans, a model which does not use any dictionary. Bleu(figure 4) scores for Banking and Regulatory domains are better for other models. These are the domains with less ambiguity i.e most of the constraints would have lower fanout.

	Regulatory	Aerospace	Banking	Combustion-T1	Combustion-T2	Structures	Medical
GoogleTrans	24.9	36	31.3	43.2	40.1	43	15.6
mBART	24.7	28.1	28.1	31.1	28.9	34.2	17
Leca	**28.4**	26.9	32.6	30.4	38.9	37.9	16.3
IndicTrans	27.3	37.3	**34.4**	43.3	41.3	47.2	15.7
DictDis	27.5	**37.9**	34	**44.9**	**43.4**	**51.3**	**20.6**

Fig. 4: BLEU scores for various datasets and approaches

These observations bring in the following questions

1. Whether dictionary is being used at all by the model?
   - Are the translations present in dictionary very common in the dataset that the model picks correct translation from the data itself rather than picking from dictionary?
2. Why the model is performing comparatively low when fanout =1?
   - For fanout 1 replacement(vs. no replacement) seems more aggressive on test sets relative to the replacement of fanout > 1, does it remain same for training sets?

To understand and answer these questions, we performed frequency analysis on the occurrence of dictionary words/phrases in the corpus.

2 Frequency Analysis of Dictionary Data in Samananthar Dataset

2.1 Problem Statement

Pondering the question why when fanout is 1, DictDis remained at the level of IndicTrans which does not use any dictionary. We want to check whether the model is getting to learn to use the dictionary and pick constraints from it at all. We check the frequency of occurrences of dictionary words/phrases in the training corpus. This would show where the problem exactly is

- Distribution and diversity of the dictionary dataset and training corpus used
- (or) The model itself

2.2 Processing

Following quantities are calculated (code) on and the training corpus, Samananthar and the generalised en-hi dictionary used for training

1. No. of LHS words/phrases of dictionary with non-zero occurrences in corpus
2. Frequency of LHS words/phrases in the source sentences (call it "LHS freq")
3. Frequency of LHS words/phrases in the source sentences when RHS translations occur in corresponding target sentences (call it "LRHS freq")

2.3 Results and Observations

2.3.1

Number of words/phrases in the dictionary = 11,767
Among these 11,593 have non-zero occurrences and 7,799 i.e approximately 75% have frequency greater than 100. Large fraction of the dictionary do appear in the source sentences. Hence, we can guarantee that learning to use the dictionary is possible from this dataset.

2.3.2

For the entire corpus and dictionary

LHS freq = 2,90,75,364
LRHS freq = 82,74,281
Constraint copy rate = 1 : 3 (approx)

i.e. for every 3 constraints given by the dictionary 1 constraint is copied into the translation in the training corpus. This is a fair distribution of constraints being copied. Therefore, the dictionary translations are not too common in the data. Model should be able to learn to use the dictionary with this data.

2.3.3

For the words/phrases with fanout = 1

LHS freq = 20,96,848
LRHS freq = 6,27,896
Constraint copy rate = 1 : 3.3 (approx)

For the words/phrases with fanout > 1
LHS freq = 2,69,78,516
LRHS freq = 76,46,385
Constraint copy rate = 1 : 3.5 (approx)

Constraint copying is equally aggressive for both the cases of fanout =1 and > 1, unlike test sets where ratios are even more skewed. This could partly explain the performance issues of fanout =1.
Word-by-word frequencies and analysis can be found here

3 Improving DictDis with Knowledge Distillation

We have kept some initial efforts in improving DictDis using knowledge-distillation methods by creating a student-teacher model. The teacher model would use dictionaries to add the lexical constraints and the student model learns how to disambiguate and select those constraints from the teacher. We wanted to follow the AMAL : Adaptive Mixing of Auxiliary Losses setting.

We have kept it on hold owing to few constraints. If we have a student model for learning disambiguation and copy part, then we have 2 large same-sized models for teacher and student. This makes the training quite time-taking. It also seemed unnecessary as the model itself is deep and copy part is being able learn here itself. Also the process of inference remains in questions as teacher model would be discarded that actually provides the constraints to disambiguate.

4 Lexicon Selection for Lexically Constrained NMT

4.1 Problem Statement

Currently user is asked to select the dictionaries to add lexical constraints from to the translations. We want to make this step automatic. Based on the source text, model should be able to infer the domain and accordingly select the relevant dictionaries to add constraints. Given a set of dictionaries and source text, we aim to select a minimal subset of dictionaries that covers most of the source text.

4.2 Approaches Summary

As a first step, we aim to create a baseline that involves no semantics i.e when we check for coverage of the source by a dictionary, we use simple syntactical token match after lemmatisation instead of matching semantically.

We do not expect constraints of common words to get added. Such words are of least importance and become noise when finding the coverage. To avoid this, we do the following

1. Remove stop words as a pre-processing step after tokenisation
2. Use idfs calculated on Samananthar corpus as weights while finding coverage

4.3 Weighted Overlap

4.3.1 Formulation

A similarity score is calculated for each dictionary. Using a threshold on this similarity score, the top dictionaries are give as the selected ones. This similarity score captures the coverage by calculating number of tokens in the source that occur in the dictionary weighted by idf of the token indicating its uniqueness of appearance.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ corpus}$$

For a phrasal match, we add number of words in phrase instead of single match.

4.3.2 Results

The model is run on several Wikipedia articles and its results (figure 5) are analysed qualitatively. A mixture of dictionaries are being selected - few domain related, general large sized(like chem, defence, eng-hin-gen) along with some noise. But domain-specific dictionaries are not at all picked up. For example, for the article on Dinosaur, zoology is getting selected but palaeontology isn't.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
electrical.csv	electrical.csv	electrical.csv	electrical.csv
chem.csv	nematology.csv	aerospace_glossary.csv	med_comp.csv
aerospace_glossary.csv	aerospace_glossary.csv	geography.csv	aerospace_glossary.csv
geography.csv	geophysics.csv	chem.csv	nematology.csv
nematology.csv	geography.csv	med_comp.csv	geophysics.csv
med_comp.csv	it.csv	it.csv	chem.csv
it.csv	itGlossary.csv	itGlossary.csv	it.csv
itGlossary.csv	med_comp.csv	nematology.csv	itGlossary.csv
geophysics.csv	broadcasting.csv	defence.csv	geography.csv
broadcasting.csv	chem.csv	geophysics.csv	zoology.csv
defence.csv	history.csv	broadcasting.csv	broadcasting.csv
mining_geo.csv	defence.csv	climatology.csv	med.csv
med.csv	itihas_paribhasha.csv	administrative.csv	clinical.csv

Fig. 5: Preference order of dictionaries for Wiki articles given by weighted overlap

4.4 Weighted Overlap with dictionary size normalization

4.4.1 Formulation

For some dictionaries, the size itself can be very large and general, due to which they might achieve undesirably higher similarity score in the previous formulation. To overcome this, a new formulation that performs normalization with dictionary size is used.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ dictionary}$$

4.4.2 Results

The results (figure 6) were very poor. The dictionary size normalisation seems to be overdoing. All the small sized dictionaries were pushed to top leaving no relevant dictionaries in the selection.

4.5 Observations and next steps

We found results of weighted overlap methods unsatisfying. Moreover, it does not model the set cover we desire. Set cover implies subset of dictionaries that together try to cover the entire source. Unlike, weighted overlap

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv
adhoc.csv	geophysics.csv	geophysics.csv	geophysics.csv
geophysics.csv	nematology.csv	nematology.csv	nematology.csv
nematology.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
eng-hindi.csv	adhoc.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi-general.csv	eng-hindi-general.csv	it.csv	it.csv
it.csv	it.csv	itGlossary.csv	itGlossary.csv
itGlossary.csv	itGlossary.csv	broadcasting.csv	broadcasting.csv
broadcasting.csv	itihas_paribhasha.csv	NGMA_Dict.csv	palaeobotany.csv
western_music.csv	broadcasting.csv	maanachitravijnan.csv	western_music.csv
NGMA_Dict.csv	NGMA_Dict.csv	western_music.csv	adhoc.csv
palaeobotany.csv	sociology.csv	kosh_vigyan.csv	NGMA_Dict.csv
agri.csv	western_music.csv	phy.csv	palaeontology.csv
economic_geo.csv	history.csv	physicsGlossary.csv	phy_mal.csv
kosh_vigyan.csv	palaeobotany.csv	administrative.csv	phy.csv

Fig. 6: Preference order of dictionaries for Wiki articles given by weighted overlap with dictionary size normalisation

- Finds amount of overlap of a dictionary independent of the other dictionaries
- Selection of a subset of dictionaries is done based on a given threshold i.e. a minimum amount of overlap required which would give the best dictionaries in terms of "a dictionary alone being able to represent the source"

We hence moved to use the subset selecting sub-modlar functions provided by submidlib. They assign a score for a subset. Based on a given budget i.e. size of subset, the optimal one is found.

4.6 Problem with Graph-cut

For covering a set U, by a subset X of V, Graph-cut function is defined as

$$f_{gc}(X) = \sum_{i \in U, j \in X} s_{ij} - \lambda \sum_{i,j \in X} s_{ij} \tag{4}$$

Matching it to our problem setting

- U is the source text, then i could be modeled as a token
- X is the subset of dictionaries, then j is a dictionary in the subset

Hence, s_{ij} is similarity between a word and an entire dictionary which is not we want to model in our problem. Therefore graph-cut is not the right choice.

4.7 Set Cover

4.7.1 Formulation

It is an implementation of the regular set cover. For covering a set of concepts U, by a subset A of V, its Set Cover evaluation is defined as

$$f(A) = w(\cup_{a \in A} \gamma(a)) = w(\gamma(A)) \tag{5}$$

where $\gamma(A)$ refers to the set of concepts covered by A and $w(\gamma(A))$ is total weight of concepts covered by elements in A

Matching it to our problem setting, the concepts to be covered are tokens in the source, A is the subset of dictionaires, $\gamma(a)$ is the number of tokens of source that occur in dictionary a and w is the idf weighted coverage.

4.7.2 Results

The results (figure 7) are satisfying for a baseline. Domain specific dictionaries are coming in the top. For example maanachitravijnan for Topographic map. General large sized dictionaries(like chem, defence, eng-hin-gen) are coming along with due to their large coverage of the source.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
mining_geo.csv	geography.csv	geography.csv	zoology.csv
chem.csv	electrical.csv	sociology.csv	geography.csv
geography.csv	defence.csv	physicsGlossary.csv	palaeobotany.csv
climatology.csv	med_comp.csv	maanachitravijnan.csv	med_comp.csv
petrology.csv	petrology.csv	chem.csv	chem.csv
sociology.csv	sociology.csv	geology.csv	defence.csv
med_comp.csv	hom_sci.csv	broadcasting.csv	history.csv
geology.csv	itihas_paribhasha.csv	linguistics.csv	bio.csv
bio.csv	electronics.csv	history.csv	botany.csv
mineralogy.csv	lib_info.csv	chem_eng.csv	sociology.csv
chem_eng.csv	clinical.csv	defence.csv	clinical.csv
palaeontology.csv	zoology.csv	psychology.csv	hom_sci.csv
history.csv	quality_cont.csv	med_comp.csv	palaeontology.csv
linguistics.csv	history.csv	bio.csv	electronics.csv

Fig. 7: Preference order of dictionaries for Wiki articles given by Set Cover

Code for all the above implementations can be found here

4.8 Deployment

The SetCover implementation is deployed on the AICTE interfaces (1, 2 and 3)

A challenge in the process of deploying is the position of dictionary selection in the entire pipeline. The selection is to be performed after OCR. Current OCR is highly sophisticated and takes considerable amount of time for large books. Hence providing a preference order of dictionaries and then asking the user to perform selection requires user to intervene in the process after a considerable time.

To overcome this issue, as a current workaround we ask user to give number of dictionaries to auto-select and use them automatically further down the pipeline.

Work is being done on giving user a preference order of dictionaries to select from. This we currently add only for the machine-readable sources on which simple-quick OCR is done to obtain the text.

Code can be found here

5 Future Work

5.1 Extension for multilingual NMT

Current implementation performs pre-processing considering the source language is english. This has to be generalised by performing language-specific pre-processing

Currently subsetting is done from all the available dictionaries. In case of multilingual NMT, there would be dictionaries of multiple source and translation languages. Subsetting should be done only from those dictionaried corresponding to the specified source and translation languages.

5.2 Analysis of current Implementation

Setup profiling feedback on the interface for dictionary selection. Come up with measures that can quantify the results of this functionality.

5.3 Setting up a Learning Problem

Develop the model as a learning problem where we learn the weights of a mixture of multiple submodular and modular functions using max margin as in here.

We seem to need a mixture of different submodular set coverage functions (with normalization, without normalization, with idf, without idf) and several modular functions (quality of each dicitonary, quality of the words being covered - say nouns vs. verbs) and learning mixtures of those

5.4 Keyword based coverage

Translating books is one of the major aims of the pipeline. Books are a structured data, with contents and keywords providing the major information about it. Hence, finding coverage based on just contents/titles/sections/keywords instead of the entire book could work well.

This idea can be incorporated separately to make the subset selection fast. It can also be included in the multiple mixture components in the learning setting.

BTP-I

Ilindra Sai Lakshmi Shreya(190050050)
Guide: Prof. Ganesh Ramakrishnan

Indian Institute of Technology Bombay

1 Study on DictDis

1.1 What is DictDis?

DictDis : Dictionary Constrained Disambiguation for Improved NMT is a lexically constrained NMT system that disambiguates between multiple candidate translations derived from dictionaries. Domain-specific neural machine translation can benefit highly from lexical constraints drawn from domain-specific dictionaries. Prior work has largely focused on the single candidate setting where the target word or phrase is replaced by a single constraint. But dictionaries could present multiple candidate translations for a source word/phrase on account of the polysemous nature of words. DictDis learns to disambiguate the multiple constraints that correspond to a single word/phrase.

The model is trained in a soft manner such that no constraint is forced to appear in the predicted sentence. Given a source sentence and domain specific dictionary constraints consisting of multiple source-target phrases, it can either pick the most relevant constraint or abstain from picking any constraint in order to least affect the fluency of the translation

1.2 Observations

1.2.1 Incomplete Augmented representation

Given a triplet of source sentence, constraints and target sentence (X, C, Y) where $C = \{C_i^j\}$ representing i^{th} inter-phrase constraint and j^{th} intra-phrase constraint, lexical constraints C are added into the pipeline by creating an augmented representation of source sentence and the constraints as follows

$$\hat{X} = [X, \langle sep \rangle, C_1^1, \langle isep \rangle, C_1^2, \langle sep \rangle, C_2^1, ..., C_n, \langle eos \rangle] \tag{1}$$

$\langle sep \rangle$, $\langle isep \rangle$ and $\langle eos \rangle$ are symbols indicating inter-phrase constraint separation, intra-phrase constraint separation and end of sentence respectively.

It leads to the loss of some key information that which source word/phrase does each of the constraints corresponding to. This correspondence could help very much in generating the translations. Therefore, using a representation that can capture this correspondence could help improve the model's performance.

1.2.2 Fanout-based Probability Distribution

Final distribution over target vocabulary is defined as the weighted sum of prediction probability and constraint ingestion probability as follows

$$p(y_t|(y_{<t}, \hat{X})) = g_t P_t^{pred} + (1 - g_t)(P_t^{copy} + P_t^{dis}) \tag{2}$$

P_t^{pred}, P_t^{copy} and P_t^{dis} are prediction probability, copy probability and disambiguate probability respectively.

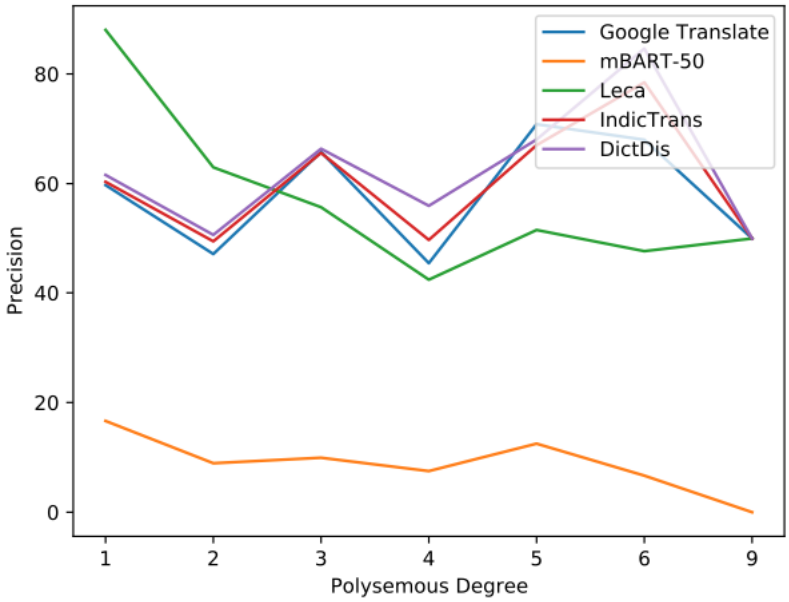

Fig. 1: SPDI for Aerospace dataset

Figures 1, 2, 3 show SPDI for various datasets obtained by various models including DictDis. Clearly the the performance of the model depends on the fanout of the constraints. Modelling fanout-based probability distribution could thererfore help improve the model's performance. One possible way could be adding fanout dependent weights. Currently work is being done in this direction, differentiating the cases of fanout =1 and fanout > 1 as follows

Disamb component :
Case 1 when $f_{t=1} : (1 - g_t)w(p_t^{copy} + p_t^{dis})$
Case 2 when $f_{t>1} : (1 - g_t)(1 - w)(p_t^{copy} + p_t^{dis})$

$$p(y_t|y_{<t}, X) = Disamb\ Component + g_t p_t^{predict} \tag{3}$$

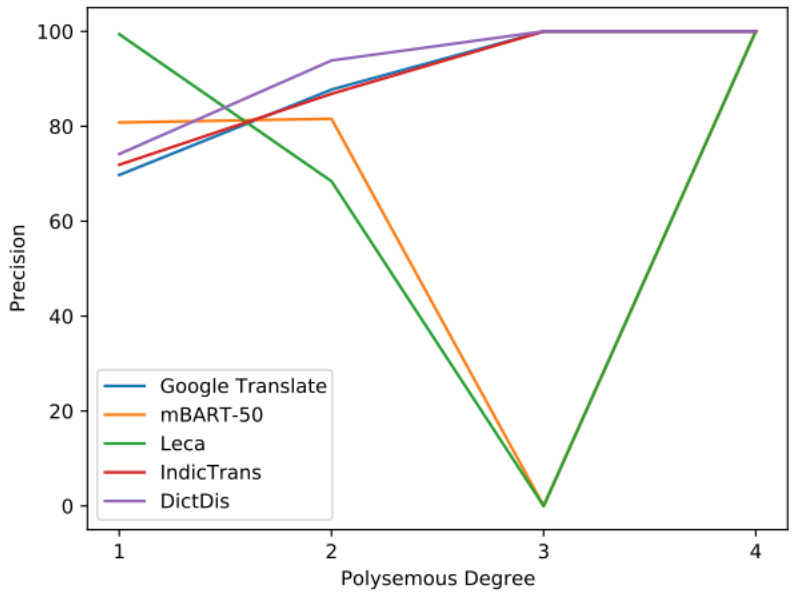

Fig. 2: SPDI for Banking dataset

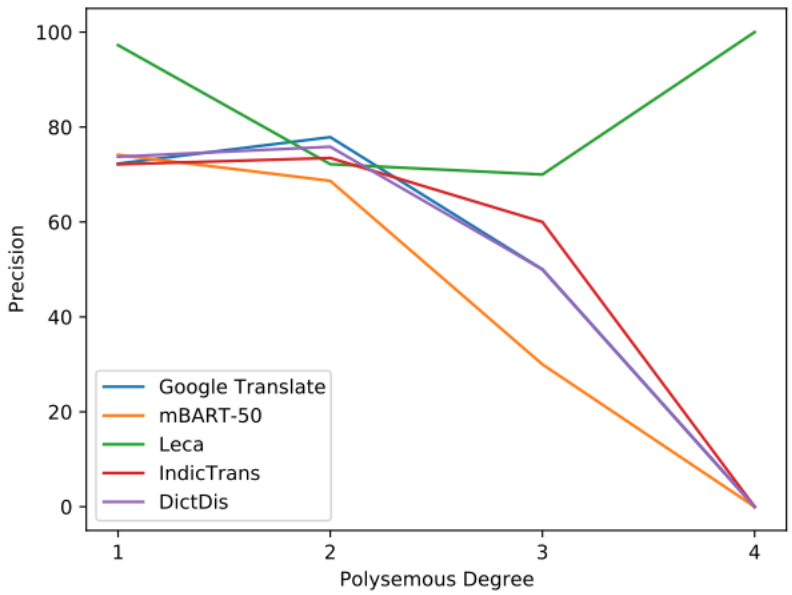

Fig. 3: SPDI for Regulatory dataset

1.2.3 Dictionary Usage by the Model

SPDI(figures 1, 2, 3) for lower fanout, especially at 1 in any domain is lower for DictDis compared to Leca. It is at the level of IndicTrans, a model which does not use any dictionary. Bleu(figure 4) scores for Banking and Regulatory domains are better for other models. These are the domains with less ambiguity i.e most of the constraints would have lower fanout.

	Regulatory	Aerospace	Banking	Combustion-T1	Combustion-T2	Structures	Medical
GoogleTrans	24.9	36	31.3	43.2	40.1	43	15.6
mBART	24.7	28.1	28.1	31.1	28.9	34.2	17
Leca	**28.4**	26.9	32.6	30.4	38.9	37.9	16.3
IndicTrans	27.3	37.3	**34.4**	43.3	41.3	47.2	15.7
DictDis	27.5	**37.9**	34	**44.9**	**43.4**	**51.3**	**20.6**

Fig. 4: BLEU scores for various datasets and approaches

These observations bring in the following questions

1. Whether dictionary is being used at all by the model?
   - Are the translations present in dictionary very common in the dataset that the model picks correct translation from the data itself rather than picking from dictionary?
2. Why the model is performing comparatively low when fanout =1?
   - For fanout 1 replacement(vs. no replacement) seems more aggressive on test sets relative to the replacement of fanout > 1, does it remain same for training sets?

To understand and answer these questions, we performed frequency analysis on the occurrence of dictionary words/phrases in the corpus.

2 Frequency Analysis of Dictionary Data in Samananthar Dataset

2.1 Problem Statement

Pondering the question why when fanout is 1, DictDis remained at the level of IndicTrans which does not use any dictionary. We want to check whether the model is getting to learn to use the dictionary and pick constraints from it at all. We check the frequency of occurrences of dictionary words/phrases in the training corpus. This would show where the problem exactly is

- Distribution and diversity of the dictionary dataset and training corpus used
- (or) The model itself

2.2 Processing

Following quantities are calculated (code) on and the training corpus, Samananthar and the generalised en-hi dictionary used for training

1. No. of LHS words/phrases of dictionary with non-zero occurrences in corpus
2. Frequency of LHS words/phrases in the source sentences (call it "LHS freq")
3. Frequency of LHS words/phrases in the source sentences when RHS translations occur in corresponding target sentences (call it "LRHS freq")

2.3 Results and Observations

2.3.1

Number of words/phrases in the dictionary = 11,767
Among these 11,593 have non-zero occurrences and 7,799 i.e approximately 75% have frequency greater than 100. Large fraction of the dictionary do appear in the source sentences. Hence, we can guarantee that learning to use the dictionary is possible from this dataset.

2.3.2

For the entire corpus and dictionary

LHS freq = 2,90,75,364
LRHS freq = 82,74,281
Constraint copy rate = 1 : 3 (approx)

i.e. for every 3 constraints given by the dictionary 1 constraint is copied into the translation in the training corpus. This is a fair distribution of constraints being copied. Therefore, the dictionary translations are not too common in the data. Model should be able to learn to use the dictionary with this data.

2.3.3

For the words/phrases with fanout = 1

LHS freq = 20,96,848
LRHS freq = 6,27,896
Constraint copy rate = 1 : 3.3 (approx)

For the words/phrases with fanout > 1
LHS freq = 2,69,78,516
LRHS freq = 76,46,385
Constraint copy rate = 1 : 3.5 (approx)

Constraint copying is equally aggressive for both the cases of fanout =1 and > 1, unlike test sets where ratios are even more skewed. This could partly explain the performance issues of fanout =1.
Word-by-word frequencies and analysis can be found here

3 Improving DictDis with Knowledge Distillation

We have kept some initial efforts in improving DictDis using knowledge-distillation methods by creating a student-teacher model. The teacher model would use dictionaries to add the lexical constraints and the student model learns how to disambiguate and select those constraints from the teacher. We wanted to follow the AMAL : Adaptive Mixing of Auxiliary Losses setting.

We have kept it on hold owing to few constraints. If we have a student model for learning disambiguation and copy part, then we have 2 large same-sized models for teacher and student. This makes the training quite time-taking. It also seemed unnecessary as the model itself is deep and copy part is being able learn here itself. Also the process of inference remains in questions as teacher model would be discarded that actually provides the constraints to disambiguate.

4 Lexicon Selection for Lexically Constrained NMT

4.1 Problem Statement

Currently user is asked to select the dictionaries to add lexical constraints from to the translations. We want to make this step automatic. Based on the source text, model should be able to infer the domain and accordingly select the relevant dictionaries to add constraints. Given a set of dictionaries and source text, we aim to select a minimal subset of dictionaries that covers most of the source text.

4.2 Approaches Summary

As a first step, we aim to create a baseline that involves no semantics i.e when we check for coverage of the source by a dictionary, we use simple syntactical token match after lemmatisation instead of matching semantically.

We do not expect constraints of common words to get added. Such words are of least importance and become noise when finding the coverage. To avoid this, we do the following

1. Remove stop words as a pre-processing step after tokenisation
2. Use idfs calculated on Samananthar corpus as weights while finding coverage

4.3 Weighted Overlap

4.3.1 Formulation

A similarity score is calculated for each dictionary. Using a threshold on this similarity score, the top dictionaries are give as the selected ones. This similarity score captures the coverage by calculating number of tokens in the source that occur in the dictionary weighted by idf of the token indicating its uniqueness of appearance.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ corpus}$$

For a phrasal match, we add number of words in phrase instead of single match.

4.3.2 Results

The model is run on several Wikipedia articles and its results (figure 5) are analysed qualitatively. A mixture of dictionaries are being selected - few domain related, general large sized(like chem, defence, eng-hin-gen) along with some noise. But domain-specific dictionaries are not at all picked up. For example, for the article on Dinosaur, zoology is getting selected but palaeontology isn't.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
electrical.csv	electrical.csv	electrical.csv	electrical.csv
chem.csv	nematology.csv	aerospace_glossary.csv	med_comp.csv
aerospace_glossary.csv	aerospace_glossary.csv	geography.csv	aerospace_glossary.csv
geography.csv	geophysics.csv	chem.csv	nematology.csv
nematology.csv	geography.csv	med_comp.csv	geophysics.csv
med_comp.csv	it.csv	it.csv	chem.csv
it.csv	itGlossary.csv	itGlossary.csv	it.csv
itGlossary.csv	med_comp.csv	nematology.csv	itGlossary.csv
geophysics.csv	broadcasting.csv	defence.csv	geography.csv
broadcasting.csv	chem.csv	geophysics.csv	zoology.csv
defence.csv	history.csv	broadcasting.csv	broadcasting.csv
mining_geo.csv	defence.csv	climatology.csv	med.csv
med.csv	itihas_paribhasha.csv	administrative.csv	clinical.csv

Fig. 5: Preference order of dictionaries for Wiki articles given by weighted overlap

4.4 Weighted Overlap with dictionary size normalization

4.4.1 Formulation

For some dictionaries, the size itself can be very large and general, due to which they might achieve undesirably higher similarity score in the previous formulation. To overcome this, a new formulation that performs normalization with dictionary size is used.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ dictionary}$$

4.4.2 Results

The results (figure 6) were very poor. The dictionary size normalisation seems to be overdoing. All the small sized dictionaries were pushed to top leaving no relevant dictionaries in the selection.

4.5 Observations and next steps

We found results of weighted overlap methods unsatisfying. Moreover, it does not model the set cover we desire. Set cover implies subset of dictionaries that together try to cover the entire source. Unlike, weighted overlap

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv
adhoc.csv	geophysics.csv	geophysics.csv	geophysics.csv
geophysics.csv	nematology.csv	nematology.csv	nematology.csv
nematology.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
eng-hindi.csv	adhoc.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi-general.csv	eng-hindi-general.csv	it.csv	it.csv
it.csv	it.csv	itGlossary.csv	itGlossary.csv
itGlossary.csv	itGlossary.csv	broadcasting.csv	broadcasting.csv
broadcasting.csv	itihas_paribhasha.csv	NGMA_Dict.csv	palaeobotany.csv
western_music.csv	broadcasting.csv	maanachitravijnan.csv	western_music.csv
NGMA_Dict.csv	NGMA_Dict.csv	western_music.csv	adhoc.csv
palaeobotany.csv	sociology.csv	kosh_vigyan.csv	NGMA_Dict.csv
agri.csv	western_music.csv	phy.csv	palaeontology.csv
economic_geo.csv	history.csv	physicsGlossary.csv	phy_mal.csv
kosh_vigyan.csv	palaeobotany.csv	administrative.csv	phy.csv

Fig. 6: Preference order of dictionaries for Wiki articles given by weighted overlap with dictionary size normalisation

- Finds amount of overlap of a dictionary independent of the other dictionaries
- Selection of a subset of dictionaries is done based on a given threshold i.e. a minimum amount of overlap required which would give the best dictionaries in terms of "a dictionary alone being able to represent the source"

We hence moved to use the subset selecting sub-modlar functions provided by submidlib. They assign a score for a subset. Based on a given budget i.e. size of subset, the optimal one is found.

4.6 Problem with Graph-cut

For covering a set U, by a subset X of V, Graph-cut function is defined as

$$f_{gc}(X) = \sum_{i \in U, j \in X} s_{ij} - \lambda \sum_{i,j \in X} s_{ij} \tag{4}$$

Matching it to our problem setting

- U is the source text, then i could be modeled as a token
- X is the subset of dictionaries, then j is a dictionary in the subset

Hence, s_{ij} is similarity between a word and an entire dictionary which is not we want to model in our problem. Therefore graph-cut is not the right choice.

4.7 Set Cover

4.7.1 Formulation

It is an implementation of the regular set cover. For covering a set of concepts U, by a subset A of V, its Set Cover evaluation is defined as

$$f(A) = w(\cup_{a \in A} \gamma(a)) = w(\gamma(A)) \tag{5}$$

where $\gamma(A)$ refers to the set of concepts covered by A and $w(\gamma(A))$ is total weight of concepts covered by elements in A

Matching it to our problem setting, the concepts to be covered are tokens in the source, A is the subset of dictionaires, $\gamma(a)$ is the number of tokens of source that occur in dictionary a and w is the idf weighted coverage.

4.7.2 Results

The results (figure 7) are satisfying for a baseline. Domain specific dictionaries are coming in the top. For example maanachitravijnan for Topographic map. General large sized dictionaries(like chem, defence, eng-hin-gen) are coming along with due to their large coverage of the source.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
mining_geo.csv	geography.csv	geography.csv	zoology.csv
chem.csv	electrical.csv	sociology.csv	geography.csv
geography.csv	defence.csv	physicsGlossary.csv	palaeobotany.csv
climatology.csv	med_comp.csv	maanachitravijnan.csv	med_comp.csv
petrology.csv	petrology.csv	chem.csv	chem.csv
sociology.csv	sociology.csv	geology.csv	defence.csv
med_comp.csv	hom_sci.csv	broadcasting.csv	history.csv
geology.csv	itihas_paribhasha.csv	linguistics.csv	bio.csv
bio.csv	electronics.csv	history.csv	botany.csv
mineralogy.csv	lib_info.csv	chem_eng.csv	sociology.csv
chem_eng.csv	clinical.csv	defence.csv	clinical.csv
palaeontology.csv	zoology.csv	psychology.csv	hom_sci.csv
history.csv	quality_cont.csv	med_comp.csv	palaeontology.csv
linguistics.csv	history.csv	bio.csv	electronics.csv

Fig. 7: Preference order of dictionaries for Wiki articles given by Set Cover

Code for all the above implementations can be found here

4.8 Deployment

The SetCover implementation is deployed on the AICTE interfaces (1, 2 and 3)

A challenge in the process of deploying is the position of dictionary selection in the entire pipeline. The selection is to be performed after OCR. Current OCR is highly sophisticated and takes considerable amount of time for large books. Hence providing a preference order of dictionaries and then asking the user to perform selection requires user to intervene in the process after a considerable time.

To overcome this issue, as a current workaround we ask user to give number of dictionaries to auto-select and use them automatically further down the pipeline.

Work is being done on giving user a preference order of dictionaries to select from. This we currently add only for the machine-readable sources on which simple-quick OCR is done to obtain the text.

Code can be found here

5 Future Work

5.1 Extension for multilingual NMT

Current implementation performs pre-processing considering the source language is english. This has to be generalised by performing language-specific pre-processing

Currently subsetting is done from all the available dictionaries. In case of multilingual NMT, there would be dictionaries of multiple source and translation languages. Subsetting should be done only from those dictionaried corresponding to the specified source and translation languages.

5.2 Analysis of current Implementation

Setup profiling feedback on the interface for dictionary selection. Come up with measures that can quantify the results of this functionality.

5.3 Setting up a Learning Problem

Develop the model as a learning problem where we learn the weights of a mixture of multiple submodular and modular functions using max margin as in here.

We seem to need a mixture of different submodular set coverage functions (with normalization, without normalization, with idf, without idf) and several modular functions (quality of each dicitonary, quality of the words being covered - say nouns vs. verbs) and learning mixtures of those

5.4 Keyword based coverage

Translating books is one of the major aims of the pipeline. Books are a structured data, with contents and keywords providing the major information about it. Hence, finding coverage based on just contents/titles/sections/keywords instead of the entire book could work well.

This idea can be incorporated separately to make the subset selection fast. It can also be included in the multiple mixture components in the learning setting.

BTP-I

Ilindra Sai Lakshmi Shreya(190050050)
Guide: Prof. Ganesh Ramakrishnan

Indian Institute of Technology Bombay

1 Study on DictDis

1.1 What is DictDis?

DictDis : Dictionary Constrained Disambiguation for Improved NMT is a lexically constrained NMT system that disambiguates between multiple candidate translations derived from dictionaries. Domain-specific neural machine translation can benefit highly from lexical constraints drawn from domain-specific dictionaries. Prior work has largely focused on the single candidate setting where the target word or phrase is replaced by a single constraint. But dictionaries could present multiple candidate translations for a source word/phrase on account of the polysemous nature of words. DictDis learns to disambiguate the multiple constraints that correspond to a single word/phrase.

The model is trained in a soft manner such that no constraint is forced to appear in the predicted sentence. Given a source sentence and domain specific dictionary constraints consisting of multiple source-target phrases, it can either pick the most relevant constraint or abstain from picking any constraint in order to least affect the fluency of the translation

1.2 Observations

1.2.1 Incomplete Augmented representation

Given a triplet of source sentence, constraints and target sentence (X, C, Y) where $C = \{C_i^j\}$ representing i^{th} inter-phrase constraint and j^{th} intra-phrase constraint, lexical constraints C are added into the pipeline by creating an augmented representation of source sentence and the constraints as follows

$$\hat{X} = [X, \langle sep \rangle, C_1^1, \langle isep \rangle, C_1^2, \langle sep \rangle, C_2^1, ..., C_n, \langle eos \rangle] \tag{1}$$

$\langle sep \rangle$, $\langle isep \rangle$ and $\langle eos \rangle$ are symbols indicating inter-phrase constraint separation, intra-phrase constraint separation and end of sentence respectively.

It leads to the loss of some key information that which source word/phrase does each of the constraints corresponding to. This correspondence could help very much in generating the translations. Therefore, using a representation that can capture this correspondence could help improve the model's performance.

1.2.2 Fanout-based Probability Distribution

Final distribution over target vocabulary is defined as the weighted sum of prediction probability and constraint ingestion probability as follows

$$p(y_t|(y_{<t}, \hat{X})) = g_t P_t^{pred} + (1 - g_t)(P_t^{copy} + P_t^{dis}) \tag{2}$$

P_t^{pred}, P_t^{copy} and P_t^{dis} are prediction probability, copy probability and disambiguate probability respectively.

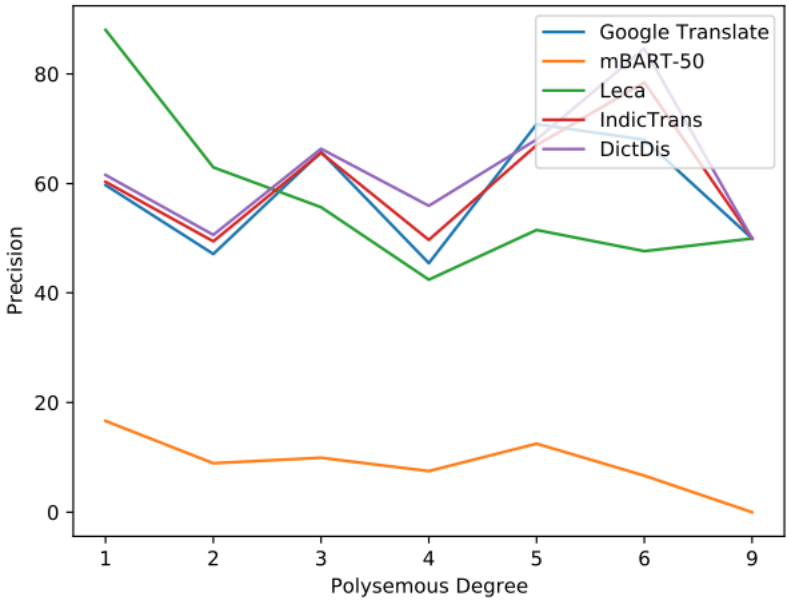

Fig. 1: SPDI for Aerospace dataset

Figures 1, 2, 3 show SPDI for various datasets obtained by various models including DictDis. Clearly the the performance of the model depends on the fanout of the constraints. Modelling fanout-based probability distribution could thererfore help improve the model's performance. One possible way could be adding fanout dependent weights. Currently work is being done in this direction, differentiating the cases of fanout =1 and fanout > 1 as follows

Disamb component :
Case 1 when $f_{t=1} : (1 - g_t)w(p_t^{copy} + p_t^{dis})$
Case 2 when $f_{t>1} : (1 - g_t)(1 - w)(p_t^{copy} + p_t^{dis})$

$$p(y_t|y_{<t}, X) = Disamb\ Component + g_t p_t^{predict} \tag{3}$$

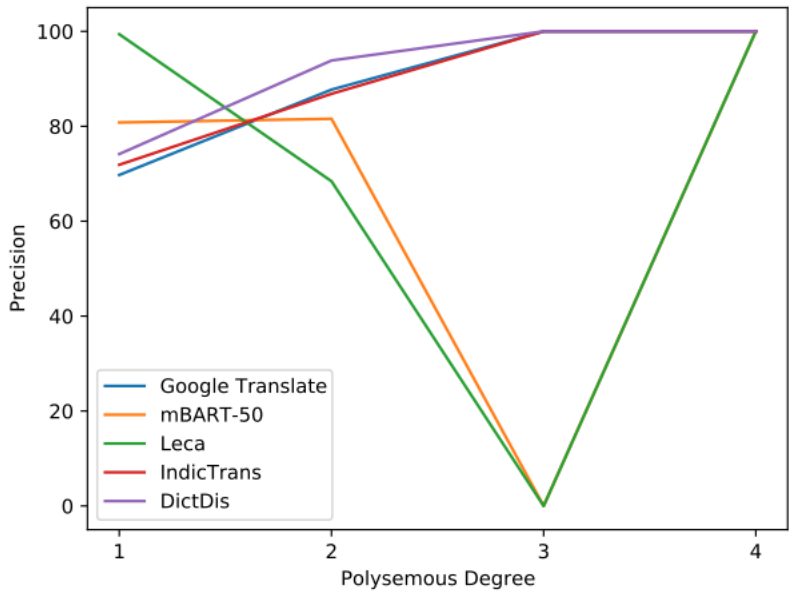

Fig. 2: SPDI for Banking dataset

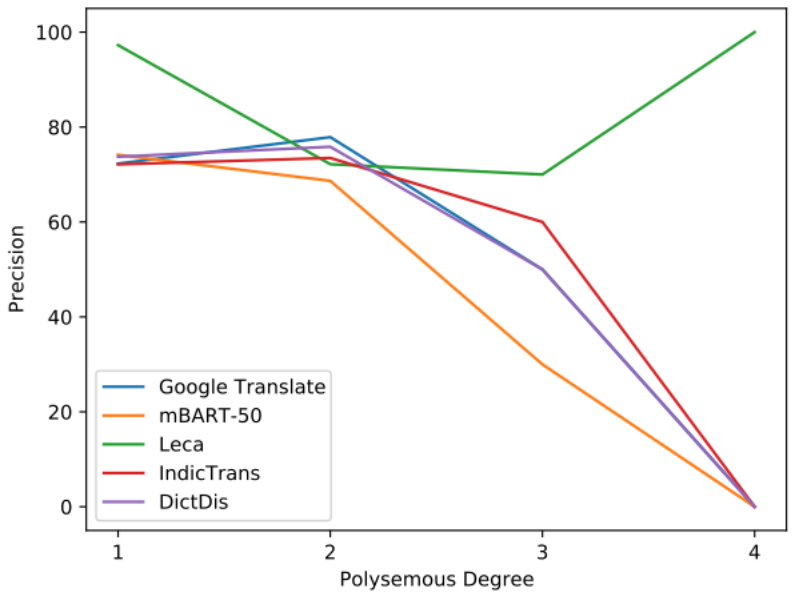

Fig. 3: SPDI for Regulatory dataset

1.2.3 Dictionary Usage by the Model

SPDI(figures 1, 2, 3) for lower fanout, especially at 1 in any domain is lower for DictDis compared to Leca. It is at the level of IndicTrans, a model which does not use any dictionary. Bleu(figure 4) scores for Banking and Regulatory domains are better for other models. These are the domains with less ambiguity i.e most of the constraints would have lower fanout.

	Regulatory	Aerospace	Banking	Combustion-T1	Combustion-T2	Structures	Medical
GoogleTrans	24.9	36	31.3	43.2	40.1	43	15.6
mBART	24.7	28.1	28.1	31.1	28.9	34.2	17
Leca	**28.4**	26.9	32.6	30.4	38.9	37.9	16.3
IndicTrans	27.3	37.3	**34.4**	43.3	41.3	47.2	15.7
DictDis	27.5	**37.9**	34	**44.9**	**43.4**	**51.3**	**20.6**

Fig. 4: BLEU scores for various datasets and approaches

These observations bring in the following questions

1. Whether dictionary is being used at all by the model?
   - Are the translations present in dictionary very common in the dataset that the model picks correct translation from the data itself rather than picking from dictionary?
2. Why the model is performing comparatively low when fanout =1?
   - For fanout 1 replacement(vs. no replacement) seems more aggressive on test sets relative to the replacement of fanout > 1, does it remain same for training sets?

To understand and answer these questions, we performed frequency analysis on the occurrence of dictionary words/phrases in the corpus.

2 Frequency Analysis of Dictionary Data in Samananthar Dataset

2.1 Problem Statement

Pondering the question why when fanout is 1, DictDis remained at the level of IndicTrans which does not use any dictionary. We want to check whether the model is getting to learn to use the dictionary and pick constraints from it at all. We check the frequency of occurrences of dictionary words/phrases in the training corpus. This would show where the problem exactly is

- Distribution and diversity of the dictionary dataset and training corpus used
- (or) The model itself

2.2 Processing

Following quantities are calculated (code) on and the training corpus, Samananthar and the generalised en-hi dictionary used for training

1. No. of LHS words/phrases of dictionary with non-zero occurrences in corpus
2. Frequency of LHS words/phrases in the source sentences (call it "LHS freq")
3. Frequency of LHS words/phrases in the source sentences when RHS translations occur in corresponding target sentences (call it "LRHS freq")

2.3 Results and Observations

2.3.1

Number of words/phrases in the dictionary = 11,767
Among these 11,593 have non-zero occurrences and 7,799 i.e approximately 75% have frequency greater than 100. Large fraction of the dictionary do appear in the source sentences. Hence, we can guarantee that learning to use the dictionary is possible from this dataset.

2.3.2

For the entire corpus and dictionary

LHS freq = 2,90,75,364
LRHS freq = 82,74,281
Constraint copy rate = 1 : 3 (approx)

i.e. for every 3 constraints given by the dictionary 1 constraint is copied into the translation in the training corpus. This is a fair distribution of constraints being copied. Therefore, the dictionary translations are not too common in the data. Model should be able to learn to use the dictionary with this data.

2.3.3

For the words/phrases with fanout = 1

LHS freq = 20,96,848
LRHS freq = 6,27,896
Constraint copy rate = 1 : 3.3 (approx)

For the words/phrases with fanout > 1
LHS freq = 2,69,78,516
LRHS freq = 76,46,385
Constraint copy rate = 1 : 3.5 (approx)

Constraint copying is equally aggressive for both the cases of fanout =1 and > 1, unlike test sets where ratios are even more skewed. This could partly explain the performance issues of fanout =1.
Word-by-word frequencies and analysis can be found here

3 Improving DictDis with Knowledge Distillation

We have kept some initial efforts in improving DictDis using knowledge-distillation methods by creating a student-teacher model. The teacher model would use dictionaries to add the lexical constraints and the student model learns how to disambiguate and select those constraints from the teacher. We wanted to follow the AMAL : Adaptive Mixing of Auxiliary Losses setting.

We have kept it on hold owing to few constraints. If we have a student model for learning disambiguation and copy part, then we have 2 large same-sized models for teacher and student. This makes the training quite time-taking. It also seemed unnecessary as the model itself is deep and copy part is being able learn here itself. Also the process of inference remains in questions as teacher model would be discarded that actually provides the constraints to disambiguate.

4 Lexicon Selection for Lexically Constrained NMT

4.1 Problem Statement

Currently user is asked to select the dictionaries to add lexical constraints from to the translations. We want to make this step automatic. Based on the source text, model should be able to infer the domain and accordingly select the relevant dictionaries to add constraints. Given a set of dictionaries and source text, we aim to select a minimal subset of dictionaries that covers most of the source text.

4.2 Approaches Summary

As a first step, we aim to create a baseline that involves no semantics i.e when we check for coverage of the source by a dictionary, we use simple syntactical token match after lemmatisation instead of matching semantically.

We do not expect constraints of common words to get added. Such words are of least importance and become noise when finding the coverage. To avoid this, we do the following

1. Remove stop words as a pre-processing step after tokenisation
2. Use idfs calculated on Samananthar corpus as weights while finding coverage

4.3 Weighted Overlap

4.3.1 Formulation

A similarity score is calculated for each dictionary. Using a threshold on this similarity score, the top dictionaries are give as the selected ones. This similarity score captures the coverage by calculating number of tokens in the source that occur in the dictionary weighted by idf of the token indicating its uniqueness of appearance.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ corpus}$$

For a phrasal match, we add number of words in phrase instead of single match.

4.3.2 Results

The model is run on several Wikipedia articles and its results (figure 5) are analysed qualitatively. A mixture of dictionaries are being selected - few domain related, general large sized(like chem, defence, eng-hin-gen) along with some noise. But domain-specific dictionaries are not at all picked up. For example, for the article on Dinosaur, zoology is getting selected but palaeontology isn't.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
electrical.csv	electrical.csv	electrical.csv	electrical.csv
chem.csv	nematology.csv	aerospace_glossary.csv	med_comp.csv
aerospace_glossary.csv	aerospace_glossary.csv	geography.csv	aerospace_glossary.csv
geography.csv	geophysics.csv	chem.csv	nematology.csv
nematology.csv	geography.csv	med_comp.csv	geophysics.csv
med_comp.csv	it.csv	it.csv	chem.csv
it.csv	itGlossary.csv	itGlossary.csv	it.csv
itGlossary.csv	med_comp.csv	nematology.csv	itGlossary.csv
geophysics.csv	broadcasting.csv	defence.csv	geography.csv
broadcasting.csv	chem.csv	geophysics.csv	zoology.csv
defence.csv	history.csv	broadcasting.csv	broadcasting.csv
mining_geo.csv	defence.csv	climatology.csv	med.csv
med.csv	itihas_paribhasha.csv	administrative.csv	clinical.csv

Fig. 5: Preference order of dictionaries for Wiki articles given by weighted overlap

4.4 Weighted Overlap with dictionary size normalization

4.4.1 Formulation

For some dictionaries, the size itself can be very large and general, due to which they might achieve undesirably higher similarity score in the previous formulation. To overcome this, a new formulation that performs normalization with dictionary size is used.

$$\text{Similarity score} = \frac{\sum_{t \in D}(freq\ of\ t\ in\ source) * idf_t}{size\ of\ dictionary}$$

4.4.2 Results

The results (figure 6) were very poor. The dictionary size normalisation seems to be overdoing. All the small sized dictionaries were pushed to top leaving no relevant dictionaries in the selection.

4.5 Observations and next steps

We found results of weighted overlap methods unsatisfying. Moreover, it does not model the set cover we desire. Set cover implies subset of dictionaries that together try to cover the entire source. Unlike, weighted overlap

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv	aerospace_glossary.csv
adhoc.csv	geophysics.csv	geophysics.csv	geophysics.csv
geophysics.csv	nematology.csv	nematology.csv	nematology.csv
nematology.csv	eng-hindi.csv	eng-hindi.csv	eng-hindi.csv
eng-hindi.csv	adhoc.csv	eng-hindi-general.csv	eng-hindi-general.csv
eng-hindi-general.csv	eng-hindi-general.csv	it.csv	it.csv
it.csv	it.csv	itGlossary.csv	itGlossary.csv
itGlossary.csv	itGlossary.csv	broadcasting.csv	broadcasting.csv
broadcasting.csv	itihas_paribhasha.csv	NGMA_Dict.csv	palaeobotany.csv
western_music.csv	broadcasting.csv	maanachitravijnan.csv	western_music.csv
NGMA_Dict.csv	NGMA_Dict.csv	western_music.csv	adhoc.csv
palaeobotany.csv	sociology.csv	kosh_vigyan.csv	NGMA_Dict.csv
agri.csv	western_music.csv	phy.csv	palaeontology.csv
economic_geo.csv	history.csv	physicsGlossary.csv	phy_mal.csv
kosh_vigyan.csv	palaeobotany.csv	administrative.csv	phy.csv

Fig. 6: Preference order of dictionaries for Wiki articles given by weighted overlap with dictionary size normalisation

- Finds amount of overlap of a dictionary independent of the other dictionaries
- Selection of a subset of dictionaries is done based on a given threshold i.e. a minimum amount of overlap required which would give the best dictionaries in terms of "a dictionary alone being able to represent the source"

We hence moved to use the subset selecting sub-modlar functions provided by submidlib. They assign a score for a subset. Based on a given budget i.e. size of subset, the optimal one is found.

4.6 Problem with Graph-cut

For covering a set U, by a subset X of V, Graph-cut function is defined as

$$f_{gc}(X) = \sum_{i \in U, j \in X} s_{ij} - \lambda \sum_{i,j \in X} s_{ij} \tag{4}$$

Matching it to our problem setting

- U is the source text, then i could be modeled as a token
- X is the subset of dictionaries, then j is a dictionary in the subset

Hence, s_{ij} is similarity between a word and an entire dictionary which is not we want to model in our problem. Therefore graph-cut is not the right choice.

4.7 Set Cover

4.7.1 Formulation

It is an implementation of the regular set cover. For covering a set of concepts U, by a subset A of V, its Set Cover evaluation is defined as

$$f(A) = w(\cup_{a \in A} \gamma(a)) = w(\gamma(A)) \tag{5}$$

where $\gamma(A)$ refers to the set of concepts covered by A and $w(\gamma(A))$ is total weight of concepts covered by elements in A

Matching it to our problem setting, the concepts to be covered are tokens in the source, A is the subset of dictionaires, $\gamma(a)$ is the number of tokens of source that occur in dictionary a and w is the idf weighted coverage.

4.7.2 Results

The results (figure 7) are satisfying for a baseline. Domain specific dictionaries are coming in the top. For example maanachitravijnan for Topographic map. General large sized dictionaries(like chem, defence, eng-hin-gen) are coming along with due to their large coverage of the source.

Sedimentary rocks	Monarchy	Topographic map	Dinosaur
eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv	eng-hindi-general.csv
mining_geo.csv	geography.csv	geography.csv	zoology.csv
chem.csv	electrical.csv	sociology.csv	geography.csv
geography.csv	defence.csv	physicsGlossary.csv	palaeobotany.csv
climatology.csv	med_comp.csv	maanachitravijnan.csv	med_comp.csv
petrology.csv	petrology.csv	chem.csv	chem.csv
sociology.csv	sociology.csv	geology.csv	defence.csv
med_comp.csv	hom_sci.csv	broadcasting.csv	history.csv
geology.csv	itihas_paribhasha.csv	linguistics.csv	bio.csv
bio.csv	electronics.csv	history.csv	botany.csv
mineralogy.csv	lib_info.csv	chem_eng.csv	sociology.csv
chem_eng.csv	clinical.csv	defence.csv	clinical.csv
palaeontology.csv	zoology.csv	psychology.csv	hom_sci.csv
history.csv	quality_cont.csv	med_comp.csv	palaeontology.csv
linguistics.csv	history.csv	bio.csv	electronics.csv

Fig. 7: Preference order of dictionaries for Wiki articles given by Set Cover

Code for all the above implementations can be found here

4.8 Deployment

The SetCover implementation is deployed on the AICTE interfaces (1, 2 and 3)

A challenge in the process of deploying is the position of dictionary selection in the entire pipeline. The selection is to be performed after OCR. Current OCR is highly sophisticated and takes considerable amount of time for large books. Hence providing a preference order of dictionaries and then asking the user to perform selection requires user to intervene in the process after a considerable time.

To overcome this issue, as a current workaround we ask user to give number of dictionaries to auto-select and use them automatically further down the pipeline.

Work is being done on giving user a preference order of dictionaries to select from. This we currently add only for the machine-readable sources on which simple-quick OCR is done to obtain the text.

Code can be found here

5 Future Work

5.1 Extension for multilingual NMT

Current implementation performs pre-processing considering the source language is english. This has to be generalised by performing language-specific pre-processing

Currently subsetting is done from all the available dictionaries. In case of multilingual NMT, there would be dictionaries of multiple source and translation languages. Subsetting should be done only from those dictionaried corresponding to the specified source and translation languages.

5.2 Analysis of current Implementation

Setup profiling feedback on the interface for dictionary selection. Come up with measures that can quantify the results of this functionality.

5.3 Setting up a Learning Problem

Develop the model as a learning problem where we learn the weights of a mixture of multiple submodular and modular functions using max margin as in here.

We seem to need a mixture of different submodular set coverage functions (with normalization, without normalization, with idf, without idf) and several modular functions (quality of each dicitonary, quality of the words being covered - say nouns vs. verbs) and learning mixtures of those

5.4 Keyword based coverage

Translating books is one of the major aims of the pipeline. Books are a structured data, with contents and keywords providing the major information about it. Hence, finding coverage based on just contents/titles/sections/keywords instead of the entire book could work well.

This idea can be incorporated separately to make the subset selection fast. It can also be included in the multiple mixture components in the learning setting.