Weakly Supervised Domain Detection

Yumo Xu and **Mirella Lapata**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
yumo.xu@ed.ac.uk, mlap@inf.ed.ac.uk

Abstract

In this paper we introduce *domain detection* as a new natural language processing task. We argue that the ability to detect textual segments that are domain-heavy (i.e., sentences or phrases that are representative of and provide evidence for a given domain) could enhance the robustness and portability of various text classification applications. We propose an *encoder-detector* framework for domain detection and bootstrap classifiers with multiple instance learning. The model is hierarchically organized and suited to multilabel classification. We demonstrate that despite learning with minimal supervision, our model can be applied to text spans of different granularities, languages, and genres. We also showcase the potential of domain detection for text summarization.

1 Introduction

Text classification is a fundamental task in Natural Language Processing (NLP) that has been found useful in a wide spectrum of applications ranging from search engines enabling users to identify content on Web sites, sentiment and social media analysis, customer relationship management systems, and spam detection. Over the past several years, text classification has been predominantly modeled as a supervised learning problem (e.g., Kim, 2014; McCallum and Nigam, 1998; Iyyer et al., 2015) for which appropriately labeled data must be collected. Such data are often domain-dependent (i.e., covering specific topics such as those relating to ''Business'' or ''Medicine'') and a classifier trained using data from one domain is likely to perform poorly on another. For example, the phrase ''*the mouse died quickly*'' may indicate negative sentiment in a customer review describing the hand-held pointing device or positive sentiment when describing a laboratory experiment performed on a rodent. The ability to handle a wide variety of domains[1] has become more pertinent with the rise of data-hungry machine learning techniques like neural networks and their application to a plethora of textual media ranging from news articles to Twitter, blog posts, medical journals, Reddit comments, and parliamentary debates (Kim, 2014; Yang et al., 2016; Conneau et al., 2017; Zhang et al., 2016).

The question of how to best deal with multiple domains when training data are available for one or few of them has met with much interest in the literature. The field of *domain adaptation* (Jiang and Zhai, 2007; Blitzer et al., 2006; Daume III, 2007; Finkel and Manning, 2009; Lu et al., 2016) aims at improving the learning of a predictive function in a *target* domain where there is little or no labeled data, using knowledge transferred from a *source* domain where sufficient labeled data are available. Another line of work (Li and Zong, 2008; Wu and Huang, 2015; Chen and Cardie, 2018) assumes that labeled data may exist for multiple domains, but in insufficient amounts to train classifiers for one or more of them. The aim of *multi-domain text classification* is to leverage all the available resources in order to improve system performance across domains simultaneously.

In this paper we investigate the question of how domain-specific data might be obtained in order to enable the development of text classification tools as well as more domain aware applications such as summarization, question answering, and

[1]The term ''domain'' has been permissively used in the literature to describe (a) a collection of documents related to a particular topic such as user-reviews in Amazon for a product category (e.g., *books, movies*), (b) a type of information source (e.g., *twitter, news articles*), and (c) various fields of knowledge (e.g., *Medicine, Law, Sport*). In this paper we adopt the latter definition of domains, although, nothing in our approach precludes applying it to different domain labels.

Transactions of the Association for Computational Linguistics, vol. 7, pp. 581–596, 2019. https://doi.org/10.1162/tacl_a_00287
Action Editor: Yusuke Miyao. Submission batch: 2/2019; Revision batch: 5/2019; Published 9/2019.

information extraction. We refer to this task as *domain detection* and assume a fairly common setting where the domains of a corpus collection are known and the aim is to identify textual segments that are domain-heavy (i.e., documents, sentences, or phrases providing evidence for a given domain).

Domain detection can be formulated as a *multilabel* classification problem, where a model is trained to recognize domain evidence at the sentence-, phrase-, or word-level. By definition then, domain detection would require training data with fine-grained domain labels, thereby increasing the annotation burden; we must provide labels for training domain detectors *and* for modeling the task we care about in the first place. In this paper we consider the problem of fine-grained domain detection from the perspective of *Multiple Instance Learning* (MIL; Keeler and Rumelhart, 1992) and develop domain models with very little human involvement. Instead of learning from individually labeled segments, our model only requires document-level supervision and optionally prior domain knowledge and learns to introspectively judge the domain of constituent segments. Importantly, we do not require document-level domain annotations either because we obtain these via distant supervision by leveraging information drawn from Wikipedia.

Our domain detection framework comprises two neural network modules; an *encoder* learns representations for words and sentences together with prior domain information if the latter is available (e.g., domain definitions), and a *detector* generates domain-specific scores for words, sentences, and documents. We obtain a segment-level domain predictor that is trained end-to-end on document-level labels using a hierarchical, attention-based neural architecture (Vaswani et al., 2017). We conduct domain detection experiments on English and Chinese and measure system performance using both automatic and human-based evaluation. Experimental results show that our model outperforms several strong baselines and is robust across languages and text genres, despite learning from weak supervision. We also showcase our model's application potential for text summarization.

Our contributions in this work are threefold; we propose domain detection, as a new fine-grained multilabel learning problem which we argue would benefit the development of domain aware NLP tools; we introduce a weakly supervised *encoder-detector* model within the context of multiple instance learning; and we demonstrate that it can be applied across languages and text genres without modification.

2 Related Work

Our work lies at the intersection of multiple research areas, including domain adaptation, representation learning, multiple instance learning, and topic modeling. We review related work below.

Domain adaptation A variety of domain adaptation methods (Jiang and Zhai, 2007; Arnold et al., 2007; Pan et al., 2010) have been proposed to deal with the lack of annotated data in novel domains faced by supervised models. Daume and Marcu (2006) propose to learn three separate models, one specific to the source domain, one specific to the target domain, and a third one representing domain general information. A simple yet effective feature augmentation technique is further introduced in Daume (2007) which Finkel and Manning (2009) subsequently recast within a hierarchical Bayesian framework. More recently, Lu et al. (2016) present a general regularization framework for domain adaptation while Camacho-Collados and Navigli (2017) integrate domain information within lexical resources. A popular approach within text classification learns features that are invariant across multiple domains while explicitly modeling the individual characteristics of each domain (Chen and Cardie, 2018; Wu and Huang, 2015; Bousmalis et al., 2016).

Similar to domain adaptation, our detection task also identifies the most discriminant features for different domains. However, whereas adaptation aims to render models more portable by transferring knowledge, detection focuses on the domains themselves and identifies the textual segments that provide the best evidence for their semantics, allowing to create data sets with *explicit* domain labels to which domain adaptation techniques can be further applied.

Multiple instance learning MIL handles problems where labels are associated with groups or *bags* of instances (documents in our case), while instance labels (segment-level domain labels) are unobserved. The task is then to make aggregate

instance-level predictions, by inferring labels either for bags (Keeler and Rumelhart, 1992; Dietterich et al., 1997; Maron and Ratan, 1998) or jointly for instances and bags (Zhou et al., 2009; Wei et al., 2014; Kotzias et al., 2015). Our domain detection model is an example of the latter variant.

Initial MIL models adopted a relatively strong consistency assumption between bag labels and instance labels. For instance, in binary classification, a bag was considered positive only if all its instances were positive (Dietterich et al., 1997; Maron and Ratan, 1998; Zhang et al., 2002; Andrews and Hofmann, 2004; Carbonetto et al., 2008). The assumption was subsequently relaxed by investigating prediction combinations (Weidmann et al., 2003; Zhou et al., 2009).

Within NLP, multiple instance learning has been predominantly applied to sentiment analysis. Kotzias et al. (2015) use sentence vectors obtained by a pre-trained hierarchical convolutional neural network (Denil et al., 2014) as features under a MIL objective that simply averages instance contributions towards bag classification (i.e., positive/negative document sentiment). Pappas and Popescu-Belis (2014) adopt a multiple instance regression model to assign sentiment scores to specific product aspects, using a weighted summation of predictions. More recently, Angelidis and Lapata (2018) propose MILNET, a multiple instance learning network model for sentiment analysis. They use an attention mechanism to flexibly weigh predictions and recognize sentiment-heavy text snippets (i.e., sentences or clauses).

We depart from previous MIL-based work in devising an encoding module with self-attention and non-recurrent structure, which is particularly suitable for modeling long documents efficiently. Compared with MILNET (Angelidis and Lapata, 2018), our approach generalizes to segments of *arbitrary* granularity; it introduces an instance scoring function that supports multilabel rather than binary classification, and takes prior knowledge into account (e.g., domain definitions) to better inform the model's predictions.

Topic modeling Topic models are built around the idea that the semantics of a document collection is governed by latent variables. The aim is therefore to uncover these latent variables—topics—that shape the meaning of the document collection. Latent Dirichlet Allocation (LDA; Blei et al. 2003) is one of the best-known topic models. In LDA, documents are generated probabilistically using a mixture over K topics that are in turn characterized by a distribution over words. And words in a document are generated by repeatedly sampling a topic according to the topic distribution and selecting a word given the chosen topic.

Although most topic models are unsupervised, some variants can also accommodate document-level supervision (Mcauliffe and Blei, 2008; Lacoste-Julien et al., 2009). However, these models are not appropriate for analyzing multiply labeled corpora because they limit documents to being associated with a single label. Multi-multinomial LDA (Ramage et al. 2009b) relaxes this constraint by modeling each document as a bag of words with a bag of labels, and topics for each observation are drawn from a shared topic distribution. Labeled LDA (L-LDA; Ramage et al., 2009a) goes one step further by directly associating labels with latent topics thereby learning label-word correspondences. L-LDA is a natural extension of both LDA by incorporating supervision and multinomial naive Bayes (McCallum and Nigam, 1998) by incorporating a mixture model (Ramage et al., 2009a).

Similar to L-LDA, DETNET is also designed to perform learning and inference in multi-label settings. Our model adopts a more general solution to the credit attribution problem (i.e., the association of textual units in a document with semantic tags or labels). Despite learning from a weak and distant signal, our model can produce domain scores for text spans of varying granularity (e.g., sentences and phrases), not just words, and achieves this with a hierarchically-organized neural architecture. Aside from learning through efficient backpropagation, the proposed framework can take incorporate useful prior information (e.g., pertaining to the labels and their meaning).

3 Problem Formulation

We formulate domain detection as a multilabel learning problem. Our model is trained on samples of document-label pairs. Each document consists of s sentences $x = \{x_1, \ldots, x_s\}$ and is associated with discrete labels $y = \{y^{(c)} | c \in [1, C]\}$. In this work, domain labels are not annotated manually but extrapolated from Wikipedia (see Section 6 for details). In a non-MIL framework, a model typically learns to predict document labels by directly conditioning

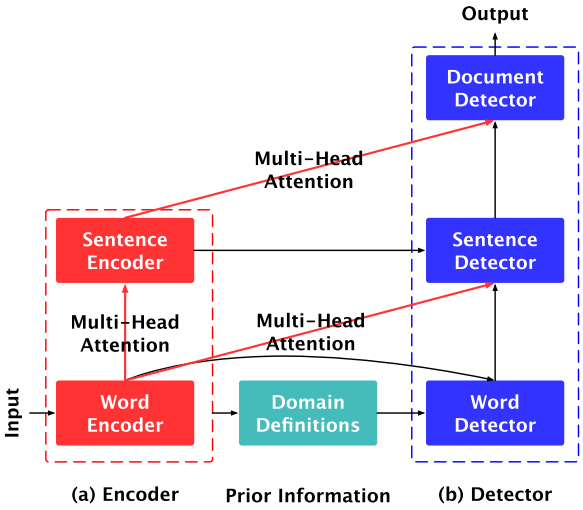

Figure 1: Overview of DETNET. The encoder learns document representations in a hierarchical fashion and the decoder generates domain scores, while selectively attending to previously encoded information. Prior information can be optionally incorporated when available at the encoding stage through parameter sharing.

on its sentence representations $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_s$ or their aggregate. In contrast, y under MIL is a learned function f_θ of *latent* instance-level labels, that is, $y = f_\theta(y_1, \ldots, y_s)$. A MIL classifier will therefore first produce domain scores for all instances (aka sentences), and then learn to integrate instance scores into a bag (i.e., document) prediction.

In this paper we further assume that the *instance-bag* relation applies to sentences and documents but also to words and sentences. In addition, we incorporate prior domain information to facilitate learning in a weakly supervised setting: Each domain is associated with a *definition* $\mathcal{U}^{(c)}$, namely, a few sentences providing a high-level description of the domain at hand. For example, the definition of the ''Lifestyle'' domain is ''the interests, opinions, behaviors, and behavioral orientations of an individual, group, or culture''.

Figure 1 provides an overview of our Domain **Det**ection **Net**work, which we call DETNET. The model includes two modules; an *encoder* learns representations for words and sentences while incorporating prior domain information; a *detector* generates domain scores for words, sentences, and documents by selectively attending to previously encoded information. We describe the two modules in more detail below.

4 The Encoder Module

We learn representations for words and sentences using identical encoders with separate learning parameters. Given a document, the two encoders implement the following steps:

$$\boldsymbol{Z}, \boldsymbol{\alpha} = \textsc{WordEnc}(\boldsymbol{X})$$
$$\boldsymbol{G} = [\boldsymbol{g}_1; \ldots; \boldsymbol{g}_s] \text{ where } \boldsymbol{g} = \boldsymbol{Z}\boldsymbol{\alpha}$$
$$\boldsymbol{H}, \boldsymbol{\beta} = \textsc{SentEnc}(\boldsymbol{G})$$

For each sentence $\boldsymbol{X} = [\boldsymbol{x}_1; \ldots; \boldsymbol{x}_n]$, the word-level encoder yields contextualized word representations $\boldsymbol{Z}$ and their attention weights $\boldsymbol{\alpha}$. Sentence embeddings $\boldsymbol{g}$ are obtained via weighted averaging and then provided as input to the sentence-level encoder, which outputs contextualized representations $\boldsymbol{H}$ and their attention weights $\boldsymbol{\beta}$.

In this work we aim to model fairly long documents (e.g., Wikipedia articles; see Section 6 for details). For this reason, our encoder builds on the Transformer architecture (Vaswani et al., 2017), a recently proposed highly efficient model that has achieved state-of-the-art performance in machine translation (Vaswani et al., 2017) and question answering (Yu et al., 2018). The Transformer aims at reducing the fundamental constraint of sequential computation that underlies most architectures based on recurrent neural networks. It eliminates recurrence in favor of applying a self-attention mechanism which directly models relationships between all words in a sentence, regardless of their position.

Self-attentive encoder As shown in Figure 2, the Transformer is a non-recurrent framework comprising m identical layers. Information on the (relative or absolute) position of each token in a sequence is represented by the use of positional encodings which are added to input embeddings (see the bottom of Figure 2). We denote position-augmented inputs in a sentence with $\boldsymbol{X}$. Our model uses four layers in both word and sentence encoders. The first three layers are identical to those in the Transformer ($m = 3$), comprising a multi-head self-attention sublayer and a position-wise fully connected feed-forward network. The last layer is simply a multi-head self-attention layer yielding attention weights for subsequent operations.

Single-head attention takes three parameters as input in the Transformer (Vaswani et al., 2017):

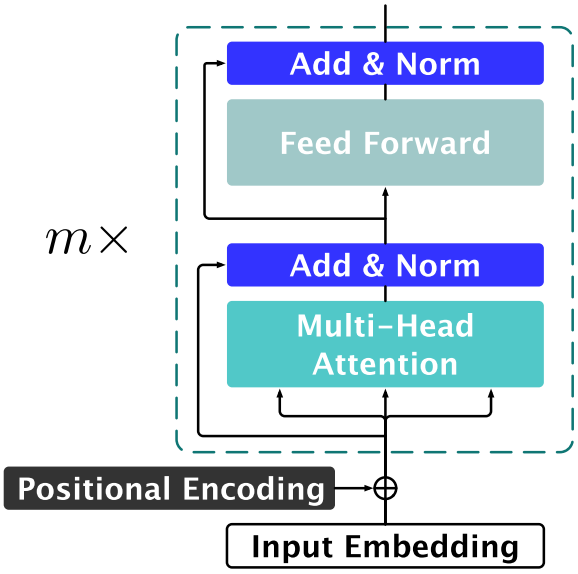

Figure 2: Self-attentive encoder in Transformer (Vaswani et al., 2017) stacking m identical layers.

a query matrix, a key matrix, and a value matrix. These three matrices are identical and equal to the inputs $\boldsymbol{X}$ at the first layer of the word encoder. The output of a single-head attention is calculated via:

$$\text{head}(\boldsymbol{X}, \boldsymbol{X}, \boldsymbol{X}) = \text{softmax}(\frac{\boldsymbol{X}\boldsymbol{X}^\intercal}{\sqrt{d_x}})\boldsymbol{X}. \quad (1)$$

Multi-head attention allows us to jointly attend to information from different representation subspaces at different positions. This is done by first applying different linear projections to inputs and then concatenating them:

$$\text{head}^{(k)} = \text{head}(\boldsymbol{X}\boldsymbol{W}_1^{(k)}, \boldsymbol{X}\boldsymbol{W}_2^{(k)}, \boldsymbol{X}\boldsymbol{W}_3^{(k)}) \quad (2)$$

$$\text{multi-head} = \text{concat}(\text{head}^{(1)}, .., \text{head}^{(r)})\boldsymbol{W}_4 \quad (3)$$

where we adopt four heads ($r = 4$) for both word and sentence encoders. The second sublayer in the Transformer (see Figure 2) is a fully-connected feed-forward network applied to each position separately and identically:[2]

$$\text{FFN}(\boldsymbol{x}) = \max(0, \boldsymbol{x}\boldsymbol{W}_5)\boldsymbol{W}_6. \quad (4)$$

After sequentially encoding input embeddings through the first three layers, we obtain contextualized word representations $\boldsymbol{Z} \in \mathbb{R}^{d_z \times n}$. Based on $\boldsymbol{Z}$, the last multi-head attention layer in the word encoder yields a set of attention matrices $\mathcal{A} = \{\boldsymbol{A}^{(k)}\}_{k=1}^r$ for each sentence where $\boldsymbol{A}^{(k)} \in \mathbb{R}^{n \times n}$. Therefore, when measuring the contributions from words to sentences (e.g., in terms of domain scores and representations) we can selectively focus on salient words within the set $\mathcal{A} = \{\boldsymbol{A}^{(k)}\}_{k=1}^r$:

$$\boldsymbol{\alpha} = \text{softmax}(\frac{1}{\sqrt{nr}}\sum_k^r\sum_\ell^n \boldsymbol{A}_{\star,\ell}^{(k)}) \quad (5)$$

where the softmax function outputs the salience distribution over words:

$$\text{softmax}(\boldsymbol{a}_\ell) = \frac{e^{\boldsymbol{a}_\ell}}{\sum_{\boldsymbol{a}_{\ell'} \in \boldsymbol{a}} e^{\boldsymbol{a}_{\ell'}}} \quad (6)$$

and obtain sentence embeddings $\boldsymbol{g} = \boldsymbol{Z}\boldsymbol{\alpha}$.

In the same vein, we adopt another self-attentive encoder to obtain contextualized sentence representations $\boldsymbol{H} \in \mathbb{R}^{d_h \times s}$. The final layer outputs multi-head attention score matrices $\mathcal{B} = \{\boldsymbol{B}^{(k)}\}_{k=1}^r$ (with $\boldsymbol{B}^{(k)} \in \mathbb{R}^{s \times s}$), and we calculate sentence salience as:

$$\boldsymbol{\beta} = \text{softmax}(\frac{1}{\sqrt{sr}}\sum_k^r\sum_j^s \boldsymbol{B}_{\star,j}^{(k)}). \quad (7)$$

Prior information In addition to documents (and their domain labels), we might have some prior knowledge about the domain, for example, its general semantic content and the various topics related to it. For example, we might expect articles from the ''Lifestyle'' domain to not talk about missiles or warfare, as these are recurrent themes in the ''Military'' domain. As mentioned earlier, throughout this paper we assume we have domain definitions $\mathcal{U}$ expressed in a few sentences as prior knowledge. Domain definitions share parameters with WordEnc and SentEnc and are encoded in a *definition* matrix $\boldsymbol{U} \in \mathbb{R}^{d_h \times C}$.

Intuitively, identifying the domain of a word might be harder than that of a sentence; on account of being longer and more expressive, sentences provide more domain-related cues than words whose meaning often relies on supporting context. We thus inject domain definitions $\boldsymbol{U}$ into our word detector only.

[2]We omit here the bias term for the sake of simplicity.

5 The Detector Module

DETNET adopts three detectors corresponding to words, sentences, and documents:

$$\boldsymbol{P} = \text{WORDDET}(\boldsymbol{Z}, \boldsymbol{U})$$
$$\boldsymbol{Q}^{instc} = [\boldsymbol{q}_1^{instc}; \ldots; \boldsymbol{q}_s^{instc}] \text{ where } \boldsymbol{q}^{instc} = \boldsymbol{P}\boldsymbol{\alpha}$$
$$\boldsymbol{Q} = \text{SENTDET}(\boldsymbol{Q}^{instc}, \boldsymbol{H})$$
$$\tilde{\boldsymbol{y}} = \text{DOCDET}(\boldsymbol{Q}, \boldsymbol{\beta})$$

WORDDET first produces word domain scores using both lexical semantic information $\boldsymbol{Z}$ and prior (domain) knowledge $\boldsymbol{U}$; SENTDET yields domain scores for sentences while integrating downstream instance signals $\boldsymbol{Q}^{instc}$ and sentence semantics $\boldsymbol{H}$; finally, DOCDET makes the final document-level predictions based on sentence scores.

Word detector Our first detector yields word domain scores. For a sentence, we obtain a *self-scoring* matrix $\boldsymbol{P}^{self}$ using its own contextual word semantic information:

$$\boldsymbol{P}^{self} = \tanh(\boldsymbol{W}_z \boldsymbol{Z}). \quad (8)$$

In contrast to the representations used in Angelidis and Lapata (2018), we generate instance scores from contextualized representations, that is, $\boldsymbol{Z}$. Because the softmax function normally favors single-mode outputs, we adopt $\tanh(\cdot) \in (-1, 1)$ as our domain scoring function to tailor MIL to our multilabel scenario.

As mentioned earlier, we use domain definitions as prior information at the word level and compute the *prior* score via:

$$\boldsymbol{P}^{prior} = \tanh(\max(\boldsymbol{0}, \boldsymbol{U}^\intercal \boldsymbol{W}_u)\boldsymbol{Z}) \quad (9)$$

where $\boldsymbol{W}_u \in \mathbb{R}^{d_u \times d_z}$ projects prior information $\boldsymbol{U}$ onto the input semantic space. The prior score matrix $\boldsymbol{P}^{prior}$ captures the interactions between domain definitions and sentential contents.

In this work, we flexibly integrate scoring components with gates, as shown in Figure 3. The key idea is to learn a *prior gate* $\boldsymbol{\Gamma}$ balancing Equations (8) and (9) via:

$$\boldsymbol{\Gamma} = \gamma\sigma(\boldsymbol{W}_{g,p}[\boldsymbol{Z}, \boldsymbol{P}^{self}, \boldsymbol{P}^{prior}]) \quad (10)$$
$$\boldsymbol{P} = \boldsymbol{\Gamma} \odot \boldsymbol{P}^{prior} + (\boldsymbol{J} - \boldsymbol{\Gamma}) \odot \boldsymbol{P}^{self} \quad (11)$$

where $\boldsymbol{J}$ is an all-ones matrix and $\boldsymbol{P} \in \mathbb{R}^{C \times n}$ is the final domain score matrix at the word-level;

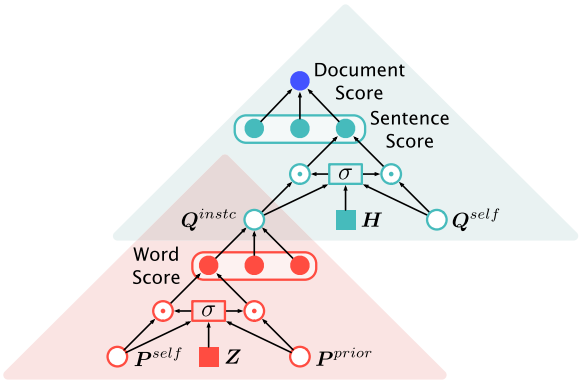

Figure 3: Domain predictions for words and sentences; the instance-bag relation applies to words-sentences (red shadow) and sentences-documents (green shadow). Squares denote representations of words or sentences, and circles are domain scores.

$\odot$ denotes element-wise multiplication and $[\cdot, \cdot]$ matrix concatenation. $\sigma(\cdot) \in (0, 1)$ is the sigmoid function and $\boldsymbol{\Gamma} \in (0, \gamma)$ the prior gate with scaling factor γ, a hyperparameter controlling the overall effect of prior information and instances.[3]

Sentence detector The second detector identifies sentences with domain-heavy semantics based on signals from the sentence encoder, prior information, and word instances. Again, we obtain a self-scoring matrix $\boldsymbol{Q}^{self}$ via:

$$\boldsymbol{Q}^{self} = \tanh(\boldsymbol{W}_h \boldsymbol{H}). \quad (12)$$

After computing sentence scores from sentence-level signals, we estimate domain scores from individual words. We do this by reusing $\boldsymbol{\alpha}$ in Equation (5), $\boldsymbol{q}^{instc} = \boldsymbol{P}\boldsymbol{\alpha}$. After gathering $\boldsymbol{q}^{instc}$ for each sentence, we obtain $\boldsymbol{Q}^{instc} \in \mathbb{R}^{C \times s}$ as the full instance score matrix.

Analogously to the word-level detector (see Equation (10)), we use a sentence-level *upward gate* $\boldsymbol{\Lambda}$ to dynamically propagate domain scores from downstream word instances to sentence bags:

$$\boldsymbol{\Lambda} = \lambda\sigma(\boldsymbol{W}_\ell[\boldsymbol{H}, \boldsymbol{Q}^{instc}, \boldsymbol{Q}^{self}]) \quad (13)$$
$$\boldsymbol{Q} = \boldsymbol{\Lambda} \odot \boldsymbol{Q}^{instc} + (\boldsymbol{J} - \boldsymbol{\Lambda}) \odot \boldsymbol{Q}^{self} \quad (14)$$

where $\boldsymbol{Q}$ is the final sentence score matrix.

Document detector Document-level domain scores are based on the sentence salience distribution $\boldsymbol{\beta}$ (see Equation (7)) and are computed as

[3] Initially, we expected to balance these effects by purely relying on the learned function without a scaling factor. This led to poor performance, however.

	Wiki-en	Wiki-zh
All Documents	31,562	26,280
Training Documents	25,562	22,280
Development Documents	3,000	2,000
Test Documents	3,000	2,000
Multilabel Ratio	10.18%	29.73%
Average #Words	1,152.08	615.85
Vocabulary Size	175,555	169,179
Synthetic Documents	200	200
Synthetic Sentences	18,922	18,312

Table 1: Statistics of Wikipedia data sets; en and zh are shorthand for English and Chinese, respectively. Synthetic documents and sentences are used in our automatic evaluation experiments discussed in Section 7.

the weighted average of sentence scores:

$$\tilde{\boldsymbol{y}} = \boldsymbol{Q}\boldsymbol{\beta}. \quad (15)$$

We use only document-level supervision for multilabel learning in C domains. Formally, our training objective is:

$$\mathcal{L} = \min -\frac{1}{N}\sum_{i}^{N}\sum_{c}^{C}\log(1+e^{-\tilde{\boldsymbol{y}}_c^{(i)}\boldsymbol{y}_c^{(i)}}) \quad (16)$$

where N is the training set size. At test time, we partition domains into a relevant set and an irrelevant set for unseen samples. Because the domain scoring function is $\tanh(\cdot) \in (-1, 1)$, we use a threshold of 0 against which $\tilde{\boldsymbol{y}}$ is calibrated.[4]

6 Experimental Set-up

Data sets DetNet was trained on two data sets created from Wikipedia[5] for English and Chinese.[6] Wikipedia articles are organized according to a hierarchy of categories representing the defining characteristics of a field of knowledge. We recursively collect Wikipedia pages by first determining the *root categories* based on their match with the domain name. We then obtain their subcategories, the subcategories of these subcategories, and so on. We treat all pages associated with a category as representative of the domain of its root category.

[4] If $\forall c \in [1, C] : \tilde{\boldsymbol{y}}_c < 0$ holds, we set $\tilde{\boldsymbol{y}}_{c^*} = 1$ and select c^* as $c^* = \arg\max_c \tilde{\boldsymbol{y}}_c$ to produce a positive prediction.

[5] http://static.wikipedia.org/downloads/2008-06/en

[6] Available at https://github.com/yumoxu/detnet

Algorithm 1 Document Generation

Input: $\mathcal{S} = \{\mathcal{S}_d\}_1^D$: Label combinations
$\mathcal{O} = \{\mathcal{O}_d\}_1^D$: Sentence subcorpora
ℓ_{max}: Maximum document length
Output: A synthetic document

```
function GENERATE(S, O, ℓ_max)
    Generate a document domain set S^doc ∈ S
    S^sent ← S^doc ∪ {GEN}
    if |S^sent| < C then ▷ Number of domain labels
        Generate a noisy domain ϵ ∈ 𝒴 \ S^sent
        S^sent ← S^sent ∪ {ϵ}
    end if
    S^cdt ← ∅;          ▷ A set of candidate domain sets
    for S_d ∈ S do
        if S_d ∈ S^sent then
            S^cdt ← S^cdt ∪ {S_d}
        end if
    end for
    n_label ← |S^cdt|          ▷ Number of unused labels
    n_sent ← ℓ_max             ▷ Number of sentence blocks
    ℒ ← ∅                      ▷ For generated sentences
    for S_d^cdt ∈ S^cdt do
        θ = min(|O_d|, n_sents + 1 − n_labels, 2n_sents / n_labels)
        Generate ℓ_d ~ Uniform(1, θ)
        Generate ℓ_d sentences ℒ_d ⊆ O_d
        ℒ ← ℒ ∪ ℒ_c
        n_sent ← ℓ_max − |ℒ|
        n_label ← n_label − 1
    end for
    ℒ ← SHUFFLE(ℒ)
    return ℒ
end function
```

In our experiments we used seven target domains: ''Business and Commerce'' (BUS), ''Government and Politics'' (GOV), ''Physical and Mental Health'' (HEA), ''Law and Order'' (LAW), ''Lifestyle'' (LIF), ''Military'' (MIL), and ''General Purpose'' (GEN). Exceptionally, GEN does not have a natural root category. We leverage Wikipedia's 12 Main Categories[7] to ensure that GEN is genuinely different from the other six domains. We used 5,000 pages for each domain. Table 1 shows various statistics on our data set.

System comparisons We constructed three variants of DetNet to explore the contribution of different model components. DetNet$_{1\mathcal{H}}$ has a single-level hierarchical structure, treating only sentences as instances and documents as bags; whereas DetNet$_{2\mathcal{H}}$ has a two-level hierarchical structure (the instance-bag relation applies to

[7] https://en.wikipedia.org/wiki/Portal:Contents/Categories.

words-sentences and sentences-documents); finally, DETNET* is our full model, which is fully hierarchical and equipped with prior information (i.e., domain definitions). We also compared DETNET to a variety of related systems, which include:

MAJOR: The Majority domain label applies to all instances.

L-LDA: Labeled LDA (Ramage et al., 2009a) is a topic model that constrains LDA by defining a one-to-one correspondence between LDA's latent topics and observed labels. This allows L-LDA to directly learn word-label correspondences. We obtain domain scores for words through the topic-word-count matrix $\boldsymbol{M} \in \mathbb{R}^{C \times V}$, which is computed during training:

$$\tilde{\boldsymbol{M}} = \frac{\boldsymbol{M}^{\intercal} + \beta}{\sum_{c}^{C} \boldsymbol{M}_{*,c}^{\intercal} + C * \beta} \tag{17}$$

where C and V are the number of domain labels and the size of vocabulary, respectively. Scalar β is a prior value set to $1/C$ and matrix $\tilde{\boldsymbol{M}} \in \mathbb{R}^{V \times C}$ consists of word scores over domains. Following the snippet extraction approach proposed in Ramage et al. (2009a), L-LDA can also be used to score sentences as the expected probability that the domain label had generated each word. For more details on L-LDA, we refer the interested reader to Ramage et al. (2009a).

HIERNET: A hierarchical neural network model described in Angelidis and Lapata (2018) that produces document-level predictions by attentively integrating sentence representations. For this model we used word and sentence encoders identical to DETNET. HIERNET does not generate instance-level predictions, however, we assume that document-level predictions apply to all sentences.

MILNET: A variant of the MIL-based model introduced in Angelidis and Lapata (2018) that considers sentences as instances and documents as bags (whereas DETNET generalizes the instance-bag relationship to words and sentences). To make MILNET comparable to our system, we use an encoder identical to DETNET—that is two Transformer encoders for words and sentences, respectively. Thus, MILNET differs from $\text{DETNET}_{1\mathcal{H}}$ in two respects: (a) word representations are simply averaged without word-level attention to build sentence embeddings and (b) context-free sentence embeddings generate sentence domain scores before being fed to the sentence encoder.

Implementation details We used 16 shuffled samples in a batch where the maximum document length was set to 100 sentences with the excess clipped. Word embeddings were initialized randomly with 256 dimensions. All weight matrices in the model were initialized with the fan-in trick (Glorot and Bengio, 2010) and biases were initialized with zero. Apart from using layer normalization (Ba et al., 2016) in the encoders, we applied batch normalization (Ioffe and Szegedy, 2015) and a dropout rate of 0.1 in the detectors to accelerate model training. We trained the model with the Adam optimizer (Kingma and Ba, 2014). We set all three gate scaling factors in our model to 0.1. Hyper-parameters were optimized on the development set. To make our experiments easy to replicate, we release our PyTorch (Paszke et al., 2017) source code.[8]

7 Automatic Evaluation

In this section we present the results of our automatic evaluation for sentence and document predictions. Problematically, for sentence predictions we do not have gold-standard domain labels (we have only extrapolated these from Wikipedia for documents). We therefore developed an automatic approach for creating silver standard domain labels which we describe below.

Test data generation In order to obtain sentences with domain labels, we exploit *lead* sentences in Wikipedia articles. Lead sentences typically define the article's subject matter and emphasize its topics of interest.[9] As most lead sentences contain domain-specific content, we can fairly confidently assume that document-level domain labels will apply. To validate this assumption, we randomly sampled 20 documents containing 220 lead sentences and asked two annotators to label these with domain labels. Annotators overwhelmingly agreed in their assignments with the document labels; the (average) agreement was $K = 0.89$ using Cohen's Kappa coefficient.

We used the lead sentences to create pseudo documents simulating real documents whose sentences cover multiple domains. To ensure that sentence labels are combined reasonably (e.g.,

[8] Available at https://github.com/yumoxu/detnet

[9] https://en.wikipedia.org/wiki/Lead_paragraph

Systems	Sentences		Documents	
	en	zh	en	zh
MAJOR	2.81†	5.99†	3.81†	4.41†
L-LDA	38.52†	37.09†	63.10†	58.74†
HIERNET	30.01†	37.26†	75.00	68.56†
MILNET	37.12†	44.37†	50.90†	69.45†
DETNET$_{1\mathcal{H}}$	47.93†	51.31†	74.91	72.85
DETNET$_{2\mathcal{H}}$	47.89†	52.50†	75.47	71.96†
DETNET*	**54.37**	**55.88**	**76.48**	**74.24**

Table 2: Performance using Macro-F$_1$% on automatically created Wikipedia test set; models with the symbol † are significantly ($p < 0.05$) different from the best system in each task using the approximate randomization test (Noreen, 1989).

MIL is not likely to coexist with LIF), prior to generating synthetic documents we traverse the training set and acquire all domain combinations $\mathcal{S}$, e.g., $\mathcal{S} = \{\{\texttt{GOV}\}, \{\texttt{GOV}, \texttt{MIL}\}\}$. We then gather lead sentences representing the same domain combinations. We generate synthetic documents with a maximum length of 100 sentences (we also clip real documents to the same length).

Algorithm 1 shows the pseudocode for document generation. We first sample document labels, then derive candidate label sets for sentences by introducing GEN and a noisy label ϵ. After sampling sentences for each domain, we shuffle them to achieve domain-varied sentence contexts. We created two synthetic data sets for English and Chinese. Detailed statistics are shown in Table 1.

Evaluation metric We evaluate system performance automatically using label-based Macro-F$_1$ (Zhang and Zhou, 2014), a widely used metric for multilabel classification. It measures model performance for each label specifically and then macro-averages the results. For each class, given a confusion matrix $\left(\begin{smallmatrix}\text{tp} & \text{fn}\\ \text{fp} & \text{tn}\end{smallmatrix}\right)$ containing the number of samples classified as true positive, false positive, true negative, and false negative, Macro-F$_1$ is calculated as $\frac{1}{C}\sum_{c=1}^{C}\frac{2\text{tp}_c}{2\text{tp}_c+\text{fp}_c+\text{fn}_c}$ where C is the number of domain labels.

Results Our results are summarized in Table 2. We first report domain detection results for documents, since reliable performance on this task is a prerequisite for more fine-grained domain detection. As shown in Table 2, DETNET does well on document-level domain detection, managing to outperform systems over which it has no clear advantage (such as HIERNET or MILNET).

As far as sentence-level prediction is concerned, all DETNET variants significantly outperform all comparison systems. Overall, DETNET* is the best system achieving 54.37% and 55.88% Macro-F$_1$ on English (en) and Chinese (zh), respectively. It outperforms MILNET by 17.25% on English and 11.51% on Chinese. The performance of the fully hierarchical model DETNET$_{2\mathcal{H}}$ is better than DETNET$_{1\mathcal{H}}$, showing positive effects of directly incorporating word-level domain signals. We also observe that prior information is generally helpful on both languages and both tasks.

8 Human Evaluation

Aside from automatic evaluation, we also assessed model performance against human elicited domain labels for sentences and words. The purpose of this experiment was threefold: (a) to validate the results obtained from automatic evaluation; (b) to evaluate finer-grained model performance at the word level; and (c) to examine whether our model generalizes to non-Wikipedia articles. For this, we created a third test set from the *New York Times*,[10] in addition to our Wikipedia-based English and Chinese data sets. For all three corpora, we randomly sampled two documents for each domain, and then from each document, we sampled one long paragraph or a few consecutive short paragraphs containing 8–12 sentences. Amazon Mechanical Turk (AMT) workers were asked to read these sentences and assign a domain based on the seven labels used in this paper (multiple labels were allowed). Participants were provided with domain definitions. We obtained five annotations per sentence and adopted the majority label as the sentence's domain label. We obtained two annotated data sets for English (Wiki-en and NYT-en) and one for Chinese (Wiki-zh), consisting of 122/14, 111/11, and 117/12 sentences/documents each.

Word-level domain evaluation is more challenging; taken out-of-context, individual words might be uninformative or carry meanings compatible with multiple domains. Expecting crowdworkers to annotate domain labels word-by-word with high confidence might be therefore problematic. In order to reduce annotation complexity, we opted for a retrieval-style task for

[10] https://catalog.ldc.upenn.edu/LDC2008T19

Systems	Sentences			Words		
	Wiki-en	Wiki-zh	NYT	Wiki-en	Wiki-zh	NYT
MAJOR	1.34†	6.14†	0.51†	1.39†	14.95†	0.39†
L-LDA	27.81†	28.94†	28.08†	24.58†	42.67	26.24
HIERNET	42.23†	29.93†	44.74†	15.57†	24.25†	18.27†
MILNET	39.30†	45.14†	29.31†	22.11†	33.10†	23.33†
DETNET$_{1\mathcal{H}}$	48.12†	51.76†	57.06†	16.21†	26.90†	21.61†
DETNET$_{2\mathcal{H}}$	54.70†	**57.60**	55.78†	**27.06**	**43.82**	26.52
DETNET*	**58.01**	51.28†	**60.62**	26.08	43.18	**27.03**

Table 3: System performance using Macro-F$_1$% (test set created via AMT); models with the symbol † are significantly ($p < 0.05$) different from the best system in each task using the approximate randomization test (Noreen, 1989).

word evaluation. Specifically, AMT workers were given a sentence and its domain label (obtained from the sentence-level elicitation study described above), and asked to highlight which words they considered consistent with the domain of the sentence. We used the same corpora/sentences as in our first AMT study. Analogously, words in each sentence were annotated by five participants and their labels were determined by majority agreement.

Fully hierarchical variants of our model (i.e., DETNET$_{2\mathcal{H}}$, DETNET*) and L-LDA are able to produce word-level predictions; we thus retrieved the words within a sentence whose domain score was above the threshold of 0 and compared them against the labels provided by crowdworkers. MILNET and DETNET$_{1\mathcal{H}}$ can only make sentence-level predictions. In this case, we assume that the sentence domain applies to *all* words therein. HIERNET can only produce document-level predictions based on which we generate sentence labels and further assume that these apply to sentence words too. Again, we report Macro-F$_1$, which we compute as $\frac{2p^*r^*}{p^*+r^*}$ where precision p^* and recall r^* are both averaged over all words.

We show model performance against AMT domain labels in Table 3. Consistent with the automatic evaluation results, DETNET variants are the best performing models on the sentence-level task. On the Wikipedia data sets, DETNET$_{2\mathcal{H}}$ or DETNET* outperform all baselines and DETNET$_{1\mathcal{H}}$ by a large margin, showing that word-level signals can indeed help detect sentence domains. Although statistical models are typically less accurate when they are applied to data that has a different distribution from the training data, DETNET* works surprisingly well on NYT, substantially outperforming all other systems. We also notice that prior information is useful in making domain predictions for NYT sentences: Because our models are trained on Wikipedia, prior domain definitions largely alleviate the genre shift to non-Wikipedia sentences. Table 4 provides a breakdown of the performance of DETNET* across domains. Overall, the model performs worst on LIF and GEN domains (which are very broad) and best on BUS and MIL (which are very narrow).

Domains	Wiki-en	Wiki-zh	NYT
BUS	78.65	68.66	77.33
HEA	42.11	81.36	64.52
GEN	43.33	37.29	43.90
GOV	80.00	37.74	62.07
LAW	69.77	41.03	46.51
LIF	17.24	27.91	50.00
MIL	75.00	65.00	80.00
Avg	58.01	51.28	60.62

Table 4: Sentence-level DETNET* performance (Macro-F$_1$%) across domains on three data sets.

With regard to word-level evaluation, DETNET$_{2\mathcal{H}}$ and DETNET* are the best systems and are significantly better against all comparison models by a wide margin, except L-LDA. The latter is a strong domain detection system at the word-level since it is able to *directly* associate words with domain labels (see Equation (17)) without resorting to document- or sentence-level predictions. However, our two-level hierarchical model is superior considering all-around performance across sentences and documents. The results here accord with our intuition from previous experiments: hierarchical models outperform simpler variants (including MILNET) because they are able to capture and exploit fine-grained domain

Domains	DETNET*	L-LDA
BUS	monopolization, enactment, panama, funding, arbitron, maturity, groceries, os, elevator, salary, organizations, pietism, contract, mercantilism, sectors	also, business, company, used, one, management, may, business, united, 2007, time, first, new, market, new
HEA	psychology, divorce, residence, pilates, dorlands, culinary, technique, emotion, affiliation, seafood, famine, malaria, oceans, characters, pregnancy	also, health, may, used, one, disease, medical, use, first, people, 1, many, time, water, care
GEN	gender, destruction, beliefs, schizophrenia, area, writers, armor, creativity, propagation, chem-informatics, overpopulation, deity, stimulation, mathematical, cosmology	also, one, theory, 1, used, time, two, may, first, example, many, called, form, would, known
GOV	penology, tenure, governance, alloys, biosecurity, authoritarianism, criticisms, burundi, motto, imperium, mesopotamia, juche, 420, krytocracy, criticism	also, government, political, state, united, party, one, minister, national, states, first, would, used, new, university
LAW	alloys, biosecurity, authoritarianism, mesopotamia, electronic, economical, pupil, pathophysiology, imperium, phonology, collusion, cantons, auctoritas, sigint, juche	law, also, united, legal, may, act, states, court, rights, one, case, state, would, v, government
LIF	teacher, freight, career, agaricomycetes, casein, manga, diplogasteria, benefit, pteridophyta, basidiomycota, ascomycota, letters, eukaryota, carcinogens, lifespan	also, used, may, often, one, made, water, food, many, use, usually, called, known, oil, time
MIL	battles, eads, insignia, commanders, artillery, width, episodes, neurasthenia, reconnaissance, elevation, freedom, length, patrol, manufacturer, demise	military, war, army, also, air, united, force, states, one, used, forces, first, royal, british, world

Table 5: Top 15 domain words in the Wiki-en development set according to DETNET* and L-LDA.

signals relatively accurately. Interestingly, prior information does not seem to have an effect on the Wikipedia data sets, but is useful when transferring to NYT. We also observe that models trained on the Chinese data sets perform consistently better than English. Analysis of the annotations provided by crowdworkers revealed that the ratio of domain words in Chinese is higher compared with English (27.47% vs. 13.86% in Wikipedia and 16.42% in NYT), possibly rendering word retrieval in Chinese an easier task.

Table 5 shows the 15 most representative domain words identified by our model (DETNET*) on Wiki-en for our seven domains. We obtained this list by weighting word domain scores $\boldsymbol{P}$ with their attention scores:

$$\boldsymbol{P}^* = \boldsymbol{P} \odot [\boldsymbol{\alpha}; \ldots; \boldsymbol{\alpha}]^\intercal \tag{18}$$

and ranking all words in the development set according to $\boldsymbol{P}^*$, separately for each domain. Because words appearing in different contexts are usually associated with multiple domains, we determine a word's ranking for a given domain based on the highest score. As shown in Table 5, *biosecurity* and *authoritarianism* are prevalent in both GOV and LAW domains. Interestingly, with contextualized word representations, fairly general English words are recognized as domain-heavy. For example, *technique* is a strong domain word in HEA and *420* in GOV (the latter is slang for the consumption of cannabis and highly associated with government regulations).

For comparison, we also show the top domain words identified by L-LDA via matrix $\tilde{\boldsymbol{M}}$ (see Equation (17)). To produce meaningful output, we have removed stop words and punctuation tokens, which are given very high domain scores by L-LDA (this is not entirely surprising since $\tilde{\boldsymbol{M}}$ is based on simple co-occurrence). Notice that no such post-processing is necessary for our model. As shown in Table 5, the top domain words identified by L-LDA (on the right) are more general and less informative, than those from DETNET* (on the left).

9 Domain-Specific Summarization

In this section we illustrate how fine-grained domain scores can be used to produce domain

summaries, following an extractive, unsupervised approach. We assume the user specifies the domains they are interested in a priori (e.g., LAW, HEA) and the system returns summaries targeting the semantics of these domains.

Specifically, we introduce DETRANK, an extension of the well-known TEXTRANK algorithm (Mihalcea and Tarau, 2004), which incorporates domain signals acquired by DETNET*. For each document, TEXTRANK builds a directed graph $G = (V, E)$ with nodes V corresponding to sentences, and undirected edges E whose weights are computed based on sentence similarity. Specifically, edge weights are represented with matrix $\boldsymbol{E}$ where each element $\boldsymbol{E}_{i,j}$ corresponds to the transition probability from vertex i to vertex j. Following Barrios et al. (2016), $\boldsymbol{E}_{i,j}$ is computed with the Okapi BM25 algorithm (Robertson et al., 1995), a probabilistic version of TF-IDF, and small weights (< 0.001) are set to zeros. Unreachable nodes are further pruned to acquire the final vertex set V.

To enhance TEXTRANK with domain information, we first multiply sentence-level domain scores $\boldsymbol{Q}$ with their corresponding attention scores:

$$\boldsymbol{Q}^* = \boldsymbol{Q} \odot [\boldsymbol{\beta}; \ldots; \boldsymbol{\beta}]^\intercal. \tag{19}$$

and for a given domain c, we can extract a (domain) sentence score vector $\boldsymbol{q}^* = \boldsymbol{Q}^*_{c,*} \in \mathbb{R}^{1\times s}$. Then, from $\boldsymbol{q}^*$, we produce vector $\tilde{\boldsymbol{q}} \in \mathbb{R}^{1\times|V|}$ representing a distribution of domain signals over sentences:

$$\hat{\boldsymbol{q}} = [\boldsymbol{q}^*_i]_{i\in V} \tag{20}$$

$$\tilde{\boldsymbol{q}} = \text{softmax}\left(\frac{\hat{\boldsymbol{q}} - \hat{\boldsymbol{q}}_{min}}{\hat{\boldsymbol{q}}_{max} - \hat{\boldsymbol{q}}_{min}}\right) \tag{21}$$

In order to render domain signals in different sentences more discernible, we scale all elements in $\hat{\boldsymbol{q}}$ to $[0, 1]$ before obtaining a legitimate distribution with the softmax function. Finally, we integrate the domain component into the original transition matrix as:

$$\tilde{\boldsymbol{E}} = \phi * \hat{\boldsymbol{q}} + (1 - \phi) * \boldsymbol{E} \tag{22}$$

where $\phi \in (0, 1)$ controls the extent to which domain-specific information influences sentence selection for the summarization task; higher ϕ will lead to summaries which are more domain-relevant. Here, we empirically set $\phi = 0.3$. The main difference between DETRANK and TEXTRANK is that TEXTRANK treats $1 - \phi$ as a damping factor and a uniform probability distribution is applied to $\hat{\boldsymbol{q}}$.

Method	Inf	Succ	Coh	All
TEXTRANK	45.45†	**51.11**	42.50†	46.35†
DETRANK	**54.55**	48.89	**57.50**	**53.65**

Table 6: Human evaluation results for summaries produced by TEXTRANK and DETRANK; proportion of times AMT workers found models Informative (Inf), Succinct (Succ), and Coherent (Coh); All is the average across ratings; symbol † denotes that differences between models are statistically significant ($p < 0.05$) using a pairwise t-test.

In order to decide which sentence to include in the summary, a node's centrality is measured using a graph-based ranking algorithm (Mihalcea and Tarau, 2004). Specifically, we run a Markov chain with $\tilde{\boldsymbol{E}}$ on G until it converges to the stationary distribution $\boldsymbol{e}^*$ where each element denotes the salience of a sentence. In the proposed DETRANK algorithm, $\boldsymbol{e}^*$ jointly expresses the importance of a sentence in the document *and* its relevance to the given domain (controlled by ϕ). We rank sentences according to $\boldsymbol{e}^*$ and select the top K ones, subject to a budget (e.g., 100 words).

We ran a judgment elicitation study on summaries produced by TEXTRANK and DETRANK. Participants were provided with domain definitions and asked to decide which summary was best according to the criteria of: *Informativeness* (does the summary contain more information about a specific domain, e.g., ''Government and Politics''?), *Succinctness* (does the summary avoid unnecessary detail and redundant information?), and *Coherence* (does the summary make logical sense?). AMT workers were allowed to answer ''Both'' or ''Neither'' in cases where they could not discriminate between summaries. We sampled 50 summary pairs from the English Wikipedia development set. We collected three responses per summary pair and determined which system participants preferred based on majority agreement.

Table 6 shows the proportion of times AMT workers preferred each system according to the criteria of Informativeness, Succinctness, Coherence, and overall. As can be seen, participants find DETRANK summaries more informative and coherent. Although it is perhaps not surprising for

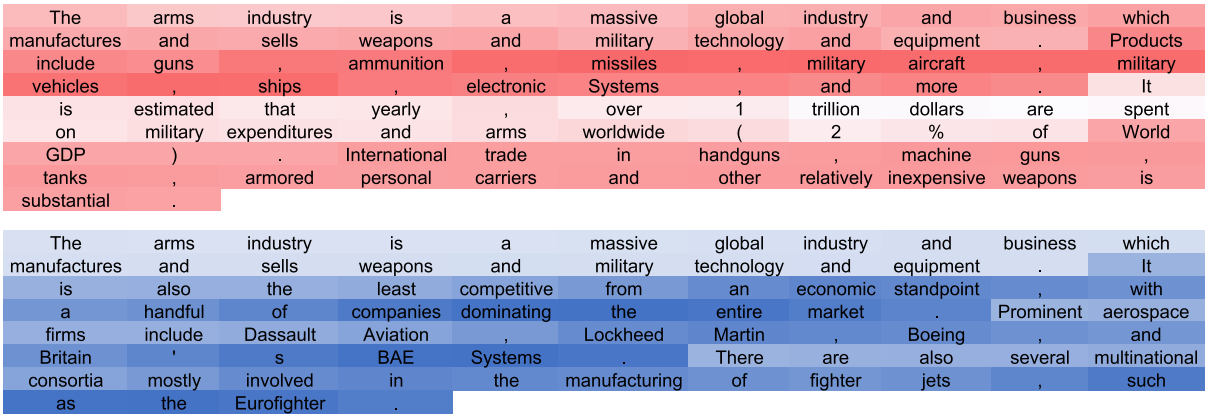

Figure 4: Summaries for the Wikipedia article ‘‘*Arms Industry*’’. The red heat map is for MIL and the blue one for BUS. Words with higher domain scores are highlighted with deeper color.

DetRank to produce summaries which are domain informative since it explicitly takes domain signals into account, it is interesting to note that focusing on a specific domain also helps discard irrelevant information and produce more coherent summaries. This, on the other hand, possibly renders DetRank's summaries more verbose (see the Succinctness ratings in Table 6).

Figure 4 shows example summaries for the Wikipedia article *Arms Industry* for domains MIL and BUS.[11] Both summaries begin with a sentence that introduces the arms industry to the reader. When MIL is the domain of interest, the summary focuses on military products such as guns and missiles. When the domain changes to BUS, the summary puts more emphasis on trade—for example, market competition and companies doing military business, such as Boeing and Eurofighter.

10 Conclusions

In this work, we proposed an encoder-detector framework for domain detection. Leveraging only weak domain supervision, our model achieves results superior to competitive baselines across different languages, segment granularities, and text genres. Aside from identifying domain-specific training data, we also show that our model holds promise for other natural language tasks, such as text summarization. Beyond domain detection, we hope that some of the work described here might be of relevance to other multilabel classification problems such as sentiment analysis (Angelidis and Lapata, 2018), relation extraction (Surdeanu et al., 2012), and named entity recognition (Tang et al., 2017). More generally, our experiments show that the proposed framework can be applied to textual data using minimal supervision, significantly alleviating the annotation bottleneck for text classification problems.

A key feature in achieving performance superior to competitive baselines is the hierarchical nature of our model, where representations are encoded step-by-step, first for words, then for sentences, and finally for documents. The framework flexibly integrates prior information which can be used to enhance the otherwise weak supervision signal or to render the model more robust across genres. In the future, we would like to investigate semi-supervised instantiations of MIL, where aside from bag labels, small amounts of instance labels are also available (Kotzias et al., 2015). It would also be interesting to examine how the label space influences model performance, especially because in our scenario the labels are extrapolated from Wikipedia and might be naturally noisy and/or ambiguous.

Acknowledgments

The authors would like to thank the anonymous reviewers and the action editor, Yusuke Miyao, for their valuable feedback. We acknowledge the financial support of the European Research Council (Lapata; award number 681760). This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via contract FA8650-17-C-9118. The views and conclusions contained herein are those of the authors and should not

[11] https://en.wikipedia.org/wiki/Arms_industry

be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation therein.

References

Stuart Andrews and Thomas Hofmann. 2004. Multiple instance learning via disjunctive programming boosting. In *Advances in Neural Information Processing Systems*, 17, pages 65–72. Curran Associates, Inc.

Stefanos Angelidis and Mirella Lapata. 2018. Multiple instance learning networks for fine-grained sentiment analysis. *Transactions of the Association for Computational Linguistics*, 6:17–31.

Andrew Arnold, Ramesh Nallapati, and William W Cohen. 2007. A comparative study of methods for transductive transfer learning. In *Proceedings of the 2007 IEEE International Conference on Data Mining*, pages 77–82. Omaha, NE.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *arXiv preprint arXiv: 1602.03606*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128. Sydney.

Konstantinos Bousmalis, George Trigeorgis, Nathan Silmerman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain separation networks. In *Advances in Neural Information Processing Systems*, 29, pages 343–351.

Jose Camacho-Collados and Roberto Navigli. 2017. Babeldomains: Large-scale domain labeling of lexical resources. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, volume 2, pages 223–228. Valencia.

Peter Carbonetto, Gyuri Dorkó, Cordelia Schmid, Hendrik Kück, and Nando De Freitas. 2008. Learning to recognize objects with little supervision. *International Journal of Computer Vision*, 77(1–3):219–237.

Xilun Chen and Claire Cardie. 2018. Multinomial adversarial networks for multi-domain text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1226–1240. New Orleans, LA.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2017. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1107–1116. Valencia.

Hal Daume III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263. Prague.

Hal Daume III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126.

Misha Denil, Alban Demiraj, Nal Kalchbrenner, Phil Blunsom, and Nando de Freitas. 2014. Modelling, visualising and summarising documents with a single convolutional neural network, University of Oxford.

Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez. 1997. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1–2):31–71.

Jenny Rose Finkel and Christopher D Manning. 2009. Hierarchical Bayesian domain adaptation. In *Proceedings of Human Language*

Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pages 602–610. Boulder, CO.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pages 249–256. Sardinia.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 448–456. Lille.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1681–1691. Beijing.

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271. Prague.

Jim Keeler and David E. Rumelhart. 1992. A self-organizing integrated segmentation and recognition neural net. In *Advances in Neural Information Processing Systems*, pages 496–503. Morgan-Kaufmann.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751. Doha.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Dimitrios Kotzias, Misha Denil, Nando De Freitas, and Padhraic Smyth. 2015. From group to individual labels using deep features. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 597–606. New York, NY.

Simon Lacoste-Julien, Fei Sha, and Michael I. Jordan. 2009. Disclda: Discriminative learning for dimensionality reduction and classification. In *Advances in Neural Information Processing Systems*, pages 897–904. Curran Associates, Inc.

Shoushan Li and Chengqing Zong. 2008. Multi-domain sentiment classification. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 257–260. Columbus, OH.

Wei Lu, Hai Leong Chieu, and Jonathan Löfgren. 2016. A general regularization framework for domain adaptation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 950–954. Austin, TX.

Oded Maron and Aparna Lakshmi Ratan. 1998. Multiple-instance learning for natural scene classification. In *Proceedings of the Annual International Conference on Machine Learning*, volume 98, pages 341–349.

Jon D. Mcauliffe and David M. Blei. 2008. Supervised topic models. In *Advances in Neural Information Processing Systems*, pages 121–128. Curran Associates, Inc.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for naive Bayes text classification. In *Proceeedings of the AAAI-98 Workshop on Learning For Text Categorization*, volume 752, pages 41–48. Madison, WI.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411. Barcelona.

Eric W. Noreen. 1989. *Computer-intensive methods for testing hypotheses*, Wiley New York.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10): 1345–1359.

Nikolaos Pappas and Andrei Popescu-Belis. 2014. Explaining the stars: Weighted multiple-instance learning for aspect-based sentiment

analysis. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 455–466. Doha.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc.

Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009a. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 248–256. Suntec.

Daniel Ramage, Paul Heymann, Christopher D. Manning, and Hector Garcia-Molina. 2009b. Clustering the tagged Web. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 54–63. Barcelona.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline M. Hancock-Beaulieu, Mike Gatford. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465. Jeju Island, Korea.

Siliang Tang, Ning Zhang, Jinjiang Zhang, Fei Wu, and Yueting Zhuang. 2017. Nite: A neural inductive teaching framework for domain specific ner. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2652–2657. Copenhagen.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010. Curran Associates, Inc.

Xiu-Shen Wei, Jianxin Wu, and Zhi-Hua Zhou. 2014. Scalable multi-instance learning. In *Proceedings of the 2014 IEEE International Conference on Data Mining*, pages 1037–1042. Shenzhen.

Nils Weidmann, Eibe Frank, and Bernhard Pfahringer. 2003. A two-level learning method for generalized multi-instance problems. In *Proceedings of the European Conference on Machine Learning*, pages 468–479. Warsaw.

Fangzhao Wu and Yongfeng Huang. 2015. Collaborative multi-domain sentiment classification. In *Proceedings of 2015 International Conference on Data Mining*, pages 459–468. Atlanic City, NJ.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489. San Diego, CA.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*.

Min-Ling Zhang and Zhi-Hua Zhou. 2014. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837.

Qi Zhang, Sally A Goldman, Wei Yu, and Jason E. Fritts. 2002. Content-based image retrieval using multiple-instance learning. In *Proceedings of the 19th International Conference on Machine Learning*, volume 2, pages 682–689. Sydney.

Ye Zhang, Iain Marshall, and Byron C. Wallace. 2016. Rationale-augmented convolutional neural networks for text classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 795–804. Austin, TX.

Zhi-Hua Zhou, Yu-Yin Sun, and Yu-Feng Li. 2009. Multi-instance learning by treating instances as non-iid samples. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1249–1256. Montreal.