Anchor Prediction: A Topic Modeling Approach

Jean Dupuy
Université de Lyon, Lyon 2, ERIC
EA3083
France
jean.dupuy@meetsys.com

Adrien Guille
Université de Lyon, Lyon 2, ERIC
EA3083
France
adrien.guille@univ-lyon2.fr

Julien Jacques
Université de Lyon, Lyon 2, ERIC
EA3083
France
julien.jacques@univ-lyon2.fr

ABSTRACT

Networks of documents connected by hyperlinks, such as Wikipedia, are ubiquitous. Hyperlinks are inserted by the authors to enrich the text and facilitate the navigation through the network. However, authors tend to insert only a fraction of the relevant hyperlinks, mainly because this is a time consuming task. In this paper we address an annotation, which we refer to as *anchor prediction*. Even though it is conceptually close to link prediction or entity linking, it is a different task that require developing a specific method to solve it. Given a source document and a target document, this task consists in automatically identifying anchors in the source document, i.e words or terms that should carry a hyperlink pointing towards the target document. We propose a contextualized relational topic model, CRTM, that models directed links between documents as a function of the local context of the anchor in the source document and the whole content of the target document. The model can be used to predict anchors in a source document, given the target document, without relying on a dictionary of previously seen mention or title, nor any external knowledge graph. Authors can benefit from CRTM, by letting it automatically suggest hyperlinks, given a new document and the set of target document to connect to. It can also benefit to readers, by dynamically inserting hyperlinks between the documents they're reading. Experiments conducted on several Wikipedia corpora (in English, Italian and German) highlight the practical usefulness of anchor prediction and demonstrate the relevancy of our approach.

CCS CONCEPTS

• **Information systems → Document topic models**.

KEYWORDS

Anchor prediction, Topic modeling, Annotation, Document network

ACM Reference Format:
Jean Dupuy, Adrien Guille, and Julien Jacques. 2022. Anchor Prediction: A Topic Modeling Approach. In *Companion Proceedings of the Web Conference 2022 (WWW '22 Companion), April 25–29, 2022, Virtual Event, Lyon, France*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3487553.3524927

1 INTRODUCTION

Wikipedia is an online and collaborative encyclopedia which is one the most visited website of the world. This attractiveness leads to the creation of nearly 600 new pages everyday in the English version of the encyclopedia[1], and other languages tend to follow this trend. In Wikipedia, articles are linked together through a network of hyperlinks. Hyperlinks allow to enrich text and facilitate horizontal reading, by giving access to additional information during the reading. Formally, a hyperlink is a directed link, from a word or term – which we refer to as *anchor* – in a source document towards a target document. A hyperlink in a Wikipedia page will look like this: `[[World Wide Web|WWW]]`. On the left of the pipe we find the title of the page to which the link points, and on the right the words that will carry the hyperlink. In Wikipedia hyperlinks are manually inserted by contributors, following a very detailed set of rules[2]. This task is community driven but could be time consuming for contributors, and some relevant anchors may remain unconnected. Thus, finding ways to improve the connectivity between pages by automatically suggesting or maybe inserting hyperlinks in the pages would facilitate the access to information and knowledge. As mentioned in [36] the use of machine learning in contributors' workflow needs to engaged the community, whose practices may differ regarding community of interest or language version. Finally, recommending anchors from a document to another specified one allow contextual hyperlink generation, which can be used by the reader to track terms relative to a specific concept during its navigation though Wikipedia.

Problem definition. Given a source document and a target document, we address the issue of automatically identifying potential anchors in the source document, given a target document. We refer to this task as *Anchor Prediction*. It is different from link prediction, where the target document isn't given, and where the locations of the anchors don't matter. It also differs from entity-linking where targets aren't documents but entries of an ontology. Finally anchor prediction is close to an annotation task, but unlike the latter it aims to identify words carrying hyperlinks between a pair of specifics document, rather than looking for probables mentions of an entire set of entities.

Use cases. Predicting anchors can be beneficial for both contributors and readers. Having wrote a new document (i.e. the source), rather than directly inserting hyperlinks, a contributor could simply specify a set of target documents, and let the system automatically locate the anchors and thus insert or recommend the hyperlinks. This set of document could also be build with any recommendation

[1]https://en.wikipedia.org/wiki/Wikipedia:Statistics
[2]https://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style/Linking

method. During a reading session, contextual hyperlinks could be automatically inserted in the document being read (i.e. the source document), pointing towards previously read documents (i.e. targets), thus helping the users contextualizing their reading.

Related work. A lot of attention has been devoted to developing means to infer latent links between documents [6, 8, 28, 44]. Prior work on this issue views a link as a connection between two entire documents, following the more general definition of link prediction in simple networks [29]. In other words, most of existing methods – including the most recent ones [9, 22, 48], with the notable exception of [7] – ignore the locations of the links, i.e. anchors, in the documents and cannot trivially be adapted to solve the anchor prediction task. Yet, they provide grounds to develop new methods suited to anchor prediction.

Other settings like automatic annotation of documents have been also studied in the last decades. An annotation method aims to find in a text the most relevant mentions of documents that belong to a predefined corpus (Wikipedia for the large majority), and returns pairs of anchors and linked documents. Those systems are build upon a wide range of machine learning methods, including statistical analysis of existing anchors [14, 35], topic modeling [21] or deep learning [15, 27]. However those methods are mostly designed to identify anchors pointing to another documents by relying on dictionaries of previously seen mentions and documents title, or knowledge graphs or onthologies.

Proposal. In this paper, we propose a new method based on a contextualized relational topic model, which we refer to as *CRTM*. We generalize the Relational Topic Model, RTM [10], itself based upon the well-known Latent Dirichlet Allocation (LDA) topic model [5]. To incorporate the network structure in LDA, RTM models undirected links between documents as a function of the topics assigned to all the words in each document. In contrast, *CRTM* models directed links between documents as a function of the topics assigned to the words surrounding the anchor (*i.e.* its context) in the source document and the topics assigned to all the words in the target document. We further improve the link function by (i) incorporating an attention mechanism to better estimate the importance of each topic for anchor prediction and (ii) incorporating additional parameters to learn new hidden representations more fitted to anchor prediction. CRTM is computationally light, language agnostic and is solely trained on texts and existing anchors, without any use of external knowledge bases nor documents titles.

Results. We assess the ability of CRTM to correctly predict anchors on several corpora made of English, Italian and German Wikipedia articles. We compare its performance with those of other variants of RTM to highlight its relevancy, and a widely used annotation tool. We also conduct an ablation study, which reveals that both our improvements to the link function are crucial to the performance of CRTM. To connect our work with prior work on link prediction, we provide additional results on this task, incorporating recent baselines. We show that, even though not initially designed to solve this task, CRTM still manages to beat some baselines, being beaten only by the most recent and specialized methods for link prediction. To conclude this paper, we show how to use CRTM in practice to predict anchors. We provide examples that show that CRTM can help in automatically detecting words or terms that should carry hyperlinks.

2 RELATED WORK

In this section we discuss the different parts of the literature we lean on. Firstly, because CRTM is a topic model, we start by reviewing prior work on topic modeling. Next, we briefly survey work on link prediction, a close task to anchor prediction. Lastly, we discuss work that leverage anchor links in other settings.

2.1 Topic Modeling

We can classify topic models in two categories: LDA-based models and neural topic models.

2.1.1 LDA-based Topic Models. These methods model documents as topic mixtures, each word in a document being assigned to one topic. LDA [5] is a generalization of pLSA [24]. It is a generative model to infer latent topics to describe a set of documents D. For each document $d \in D$, LDA infers a vector θ_d of proportions over a small number K of latent topics. The topics are characterized by a distribution β over a fixed vocabulary V. Each word occurrence $w_{d,i}$ in a document d is independently assigned a topic, encoded in a one-hot manner as $z_{d,i} \in \{0;1\}^K$, according to the distribution over words for this topic, $\beta_{\bullet,z_{d,i}}$. The Hidden Topic Markov Model (HTMM) [20] models the topics of all the words in a same document as a Markov Chain, effectively lifting the independence assumption between neighboring word occurrences. More precisely HTMM expects that all words in a same sentence share the same topic, and that consecutive sentences are more likely to have the same topic. Biterm Topic Model (BTM) [43] proposes to address a shortcoming of LDA, dealing with short texts, by modeling the generation of word co-occurrence patterns. By taking advantage of co-occurrence at the corpus-level, BTM solves the sparsity problem at the document level, which makes it suitable for short texts.

2.1.2 Neural Topic Models. With the advent of deep generative models and variational auto-encoders [25], topic modeling has been cast into the deep learning framework. NTM-R [13] relies on neural variational inference and pre-trained word embeddings, and is trained with an objective towards topic coherence. W-LDA [34] proposes to use Wasserstein auto-encoders [39] to address the problem of topic disentanglement described in [3], and enforces the Dirichlet prior. GraphBTM [50], based on the same assumption than BTM, represents biterms as a graph and uses a Graph Neural Network (GCN) [26] to extract topic mixtures from them. [49] proposes the Graph Topic Model (GTM), which extracts topic mixtures by processing a word-document graph with a GNN, to further improve topic coherence. Note that the word-document graph is distinct from an explicit document-document graph, as it solely computed from word occurrences in the documents.

It should be noted that neural topic models, such as NTM-R, W-LDA, GraphBTM or GTM, only compute topic mixtures and do not explicitly assign a topic to each word occurrence, as opposed to LDA-based models.

2.2 Link Prediction

Link prediction is a related task to the one we propose. We classify existing approaches in two categories: approaches based on topic modeling, and approaches based on node embedding.

2.2.1 Methods based on Topic Modeling. The Relation Topic Model (RTM) [10] is an extension of LDA that incorporates explicit links between documents, so that it is suitable for link prediction. Here, we quickly develop the internal functioning of this model, since CRTM generalizes it. Given a set L of explicit links between the documents in a corpus D, RTM extends LDA with the addition of a link function that models the likelihood of the links between two documents in terms of their topic proportions. This function, $\psi(y_{d,d'} = 1)$, with $y_{d,d'} \in \{0; 1\}$ a binary variable that indicates whether d and d' are linked, parametrized by $\eta, \nu \in \mathbb{R}^K$, is defined as:

$$\psi_{(d,d')}(y = 1) = \exp\left(\eta^\top\left(\overline{\mathbf{z}}_d \circ \overline{\mathbf{z}}_{d'}\right) + \nu\right), \tag{1}$$

where $\circ$ denotes the Hadamard product and $\overline{\mathbf{z}}_d = \frac{1}{N_d}\sum_i z_{d,i}$ and $\overline{\mathbf{z}}_{d'} = \frac{1}{N_d}\sum_i z_{d',i}$ represents the proportions of latent topics in document d and d' respectively. Overall, the generative process of RTM goes as follows:

- For each document d in D:
  - draw topic proportion for $d : \theta_d|\alpha \sim Dirichlet\ (\alpha)$
  - for each word occurrence $w_{d,i}$ in d:
    - draw topic assignment : $z_{d,i}|\theta_d \sim Multi\ (\theta_d)$
    - draw word : $w_{d,i}|z_{d,i} \sim Multi\ (\beta_{z_{d,i}})$
- For each pair of documents $(d, d') \in L$:
  - draw link indicator : $y_{d,n} \sim \psi\ (\cdot|\eta, \nu, z_d, z_{d'})$

$Multi(.)$ is the multinomial distribution, α the Dirichlet parameter, z_d a matrix whose rows are the $z_{d,n}$ vectors.

RTM has been modified or extended in different ways. gRTM [12] proposes to capture all pairwise topic relationships between documents and relies on a different estimation procedure based on collapsed Gibbs sampling with data augmentation. [47] proposes a non-probabilistic formulation of RTM to control the sparsity of document's topics. [46] incorporates in RTM the weighted stochastic block model [1] to identify blocks of strongly connected documents. [45] proposes a joint model that uses link structure to define clusters of documents. RTM has also triggered works in the field of Neural Topic Modeling. For instance, Relational Deep Learning (RDL) [41] is a deep hierarchical Bayesian model designed for link prediction. [2] proposes Neural Relational Topic Model (NRTM) to mimic the RTM architecture with components from deep learning. For the LDA part NRTM uses a stacked variational auto-encoder and a multi-layer perceptron to model the link function.

It should be noted that, as previously stated, neural approaches don't explicitly assign a topic to each word occurrence in a document. Thus they aren't naturally suited to anchor prediction, since they don't allow to straightforwardly extract local topical contexts.

2.2.2 Node embedding with textual information. These methods learn document embeddings in a Euclidean space and then straightforwardly predict links according to the dot-product between these embeddings. Several neural architectures have been explored to learn the embeddings from the documents' content and the network structure, such as Graph2Gauss [6], a deep energy-based encoder embedding each nodes as a Gaussian distribution, STNE, a self-translating recurrent network [28] or IDNE, an attention-based network [9]. Methods based on matrix-factorization have also been investigated, like TADW [44], an extension of the matrix formulation of DeepWalk that leverages both the network structure and the content of the documents. Similarly, GVNR-t [8], adapts the GloVe algorithm to embed networks of documents. Some methods, like RLE [17], aims at projecting linked documents in a pre-trained word embedding space.

It should be noted that all these approaches model links at the document-level, ignoring anchors (i.e. the position of the hyperlinks in the source document).

2.3 Leveraging anchors

In this last section, we discuss works that incorporate anchors to solve annotation tasks.

While some topic models take advantage of the anchor information [18, 19, 37], those methods are not suitable for a anchor prediction task. However many methods are designed to annotate documents by identifying portions of text related to another document. The first method to address this task WIKIFY [32]. TAGME [14] is an efficient method designed to annotate shorts texts with Wikipedia entries, combining three steps. For a given text, TAGME first looks for all mentions of entries in a dictionary containing titles of Wikipedia pages and anchor texts. Then an entity disambiguation step is performed using a voting scheme. The score of each mention-entity pair is computed as the sum of votes given by candidate entities of all other mentions in the text, taking into account the semantic association between two entities[31] and the probability of an entity being the link target of a given mention[30]. This step return on anchor per mention. Finally all the candidates below a certain threshold of coherence are pruned. WAT [35] is another method that extends TAGME by redefining the disambiguation step using graph-based methodologies [23]. [16] proposes a linking system adapted for Wikipedia which keep contributors in the loop. [38] is one of the first deep-learning based method designed to deal with this task, by leveraging the semantics of mention, context and entity as well as their compositionality in a unified way. BLINK [27] is another deep learning based method using a two stages approach for entity linking and annotation, based on fine-tuned BERT architectures.

3 CONTEXTUALIZED RELATIONAL TOPIC MODEL

In this section, we describe our proposal: the Contextualized Relational Topic Model, which we refer to as CRTM. It generalizes RTM so that links are modeled as a function of the topics assigned to the words in the context of the anchor in the source document and the topics assigned to all the words in the target document. The context is a set of words that surround the anchor in the text. It can be rather narrow, *e.g.* only the anchor word or the sentence in which it appears, or wider, up to the all the words in the document. In the latter case, CRTM is equivalent to RTM. We further refine the link function by (i) incorporating an attention mechanism and (ii) introducing additional parameters.

3.1 The CRTM Model

Let d and d' be two documents, and $c = \{w_i\}_{i=1}^{C}$ be the context of the link, *i.e.* a set of $C \geq 1$ words, in the source document d. The link function of *CRTM* is defined as:

$$\psi_{d,d'}(y = 1; c) = \exp\left(\eta^\top \left(Q\overline{\mathbf{z}}_{d,c} \circ Q\overline{\mathbf{z}}_{d'}\right) + \nu\right), \tag{2}$$

where $\overline{\mathbf{z}}_{d,c}$ is an attention-based weighted average of the per-word topic assignments in c and $Q \in \mathbb{R}^{K\times K}$ is a set of additional parameters. We describe these two components in the following subsections.

3.1.1 *Attention-based Weighted Average.*

Rather than calculating $\overline{\mathbf{z}}_{d,c}$ as a simple average, we rely on pre-trained word embeddings to calculate a weighted average of the topic assignments in c using an attention mechanism. Assuming a word embedding $u_i \in \mathbb{R}^p$ for each word $w_i \in V$, we compute the attention scores $s \in \mathbb{R}^k$, by measuring the scaled dot-product [40] between the embedding of the word carrying the link, u_{link} and the embedding u_j of each word w_j in c:

$$s_j = \frac{u_{\text{link}} \cdot u_j}{\sqrt{p}}. \tag{3}$$

With the attention weights $a = \text{softmax}(s)$, we calculate the weighted average:

$$\overline{\mathbf{z}}_{d,c} = \sum_{w_j \in c} a_j z_{d,j}, \tag{4}$$

where $z_{d,j}$ is the one-hot encoding of the topic assignment for the word w_j in context c. This way, the topics assigned to words that are semantically closer to the word that carries the link are given more importance.

3.1.2 *Additional Parameters.*

The form of the link function in RTM implies that linked documents should have similar topic proportions. We argue that this implicit assumption is too restricting as complementary documents that exhibit different topic proportions could be linked. Hence, we incorporate additional parameters $Q \in \mathbb{R}^{K\times K}$ so that CRTM learns new representations of the documents d and d' as linear transforms of $\overline{\mathbf{z}}_{d,c}$ and $\overline{\mathbf{z}}_{d'}$: *i.e.* $Q\overline{\mathbf{z}}_{d,c}$ and $Q\overline{\mathbf{z}}_{d'}$. This way, RTM can compare d and d' beyond simple same-topic interactions. Note that the size of Q is dictated by the estimation procedure, because it requires η to be a K-dimensional vector, which in turns enforces $Q\overline{\mathbf{z}}_{d,c}$ and $Q\overline{\mathbf{z}}_{d'}$ to be K-dimensional vectors.

3.2 Parameter Estimation

We estimate the parameters adapting the procedure in RTM, by maximizing the likelihood using the variational Expectation-Maximization algorithm [4]. The addition of Q in ψ slightly modifies the expression of the ELBO and therefore the update rules of the variational parameters of θ_d, and parameters for ψ: η and ν. The expected value of the link function becomes:

$$\mathbb{E}_q\left[\log p\left(y_{d,d'} = 1 \mid \overline{\mathbf{z}}_d, \overline{\mathbf{z}}_{d'}, \boldsymbol{\eta}\right)\right] = \boldsymbol{\eta}^T\left(Q\overline{\phi}_{d,c} \circ Q\overline{\phi}_{d'}\right) + \nu,$$

where $\phi_{d,n}$ is the variational parameter of $z_{d,n}$ and $\overline{\phi}_d = \frac{1}{N_d}\sum_n \phi_{d,n}$.

Parameters γ_d and $\phi_{d,i}$, respectively the variational parameters of θ_d and $z_{d,i}$, are updated at the E-step as follow:

$$\gamma_d = \alpha + \sum_i \phi_{d,i}, \tag{5}$$

and

$$\phi_{d,i} \propto \exp\left(\log\beta_{\cdot,w_{d,i}} + (\gamma_d) - \mathbf{1}(\mathbf{1}^T\gamma_d)) + \sum_{d'\neq d} \eta \circ \frac{Q^2\overline{\phi}_{d,i}}{N_d}\right), \tag{6}$$

with (.) the Digamma function, 1 a K-dimensional vector of ones, and N_d the number of words in document d.

During the M-step we update the model's parameters β, η, ν and Q, according to the new values of γ_d and $\phi_{d,i}$. The update rule for β didn't change from LDA:

$$\beta_{k,w} \propto \sum_d \sum_n \mathbf{1}(w_{d,n} = w)\phi_{d,n}^k. \tag{7}$$

The link function's parameters η and ν are updated as follow:

$$\eta \leftarrow \log\left(\overline{\Pi}\right) - \log\left(\overline{\Pi} + \rho\overline{\pi}_\alpha\right) - \mathbf{1}\nu, \tag{8}$$

and

$$\nu \leftarrow \log\left(L - \mathbf{1}^T\overline{\Pi}\right) - \log\left(\rho(1 - \mathbf{1}^T\overline{\pi}_\alpha) + L - \mathbf{1}^T\overline{\Pi}\right), \tag{9}$$

with $\overline{\Pi} = \sum_{(d,d')} Q\overline{\phi_d} \circ Q\overline{\phi_{d'}}$, $\overline{\pi}_\alpha = Q\frac{\alpha}{1^T\alpha} \circ Q\frac{\alpha}{1^T\alpha}$ and L the total number of observed links. ρ denote the number of negative examples needed, following the regularization procedure provided by [11]. Instead of using a $L2$ regularizer, we introduce negative observations, where $y = 0$. The observations are associated with a similarity $\overline{\pi}_\alpha$, the expected Hadamard product of any two documents given the Dirichlet prior of the model.

Finally the coefficients of Q are updated with

$$Q_{i,j} \leftarrow Q_{i,j} + l \times \sum_{(d,d'\in L)} \frac{\eta_i}{K}\left(\overline{\phi}_{d,j}\sum_{n=1}^{K} Q_{i,n}\overline{\phi}_{d',n} + \overline{\phi}_{d',j}\sum_{n=1}^{K} Q_{i,n}\overline{\phi}_{d,n}\right) \tag{10}$$

, where l is a learning rate. Note that updates to Q are strictly positive, which could cause numerical instability. While we could constrain Q to prevent its norm from monotonically increasing and rely on the Lagrange multiplier to estimate it, we choose to address this issue differently and simply normalize each row of Q with its $L2$-norm after each update. This tricks has been shown to work well in [42] for instance.

3.3 Time Complexity

Estimating the parameters of the LDA component of CRTM has a time complexity of $O(|D| \times K \times N)$, with $|D|$ the number of documents, K the number of topics and N the average number of words per document. Estimating the parameters of the relational component of CRTM has a complexity of $O(L \times K^2)$, where L is the number of observed links in the network, and K^2 is engendered by the addition of Q in the link function. The proposed model thus have an overall $O(|D| \times K \times N + L \times K^2)$ time complexity.

3.4 Relationship to gRTM

The link function of gRTM [12] incorporates a matrix $Q' \in \mathbb{R}^{K\times K}$, as follow:

$$\psi_{d,d'}(y = 1) = \sigma\left(\overline{\mathbf{z}}_d Q'\overline{\mathbf{z}}_{d'}\right), \tag{11}$$

where σ can be either the sigmoid function or the exponential function, as in RTM. This formulation arises from the fact that $\eta^\top\left(\overline{\mathbf{z}}_d \circ \overline{\mathbf{z}}_{d'}\right) = \overline{\mathbf{z}}_d\text{diag}(\eta)\overline{\mathbf{z}}_{d'}$. Rather than enforcing Q' to be a diagonal matrix to stick to RTM, gRTM considers Q' as a full matrix that allows capturing inter-topics interactions. This is quite different from what *CRTM* does, as the purpose of Q isn't to replace η but rather to project both $\overline{\mathbf{z}}_{d,c}$ and $\overline{\mathbf{z}}_{d'}$ to a new comparison space to improve link prediction.

4 EXPERIMENTS

First, we quantitatively evaluate CRTM against RTM, TAGME and variants of RTM on the task of anchor prediction. Then we highlight the importance of contextualizing the link function, and show that both refinements we bring to the function are crucial to the performance of CRTM. Then we evaluate CRTM on a link prediction task against RTM and recent baselines. This experiment is less a way to evaluate if our model can reach state of the art results, since CRTM is designed for anchor prediction, than assess if our modifications leads to a weaker generalization. Lastly, we illustrate how to use our model for anchor prediction in practice.

4.1 Datasets

To demonstrate the relevancy of our proposal we construct six datasets using Wikipedia. Each dataset is made of introductory sections of articles belonging to a specific category. Articles are sampled in a snow ball fashion following the links, starting from the main page of the category. Furthermore, since we have to construct our own datasets, we can easily evaluate our work on different languages, while commonly used datasets tend to be in only in English. We consider three languages: English, Italian and German. For each language we extract a dataset related to the main category about physics (*Physics, Fisica* and *Physik*), and society (*Society, Società* and *Gesellschaft*). We choose these categories to take into account different editorial policies, writing styles and vocabularies. Table 1 summarizes the general properties of these datasets.

4.2 Tasks and metrics

We consider two tasks. The first is an anchor prediction task, and consists in locating the words carrying hyperlinks, hidden during training, given a link between a source and a target documents. Based on the scores given by $\psi_{d,d'}$, we:

- Calculate the precision at n, *i.e.* the fraction of hidden links for which the word carrying the hyperlink is ranked among the top n words.
- Construct the ROC curve, to visually assess the overall capacity of the models to give high ranks to the word carrying the hidden hyperlinks.

The second is a link prediction task. We hide a percent of edges and compare the cosine similarity between hidden pairs and negative examples of unconnected documents. We report Area Under the ROC Curve.

4.3 Methodology

4.3.1 Anchor Prediction. We hide one link per document, which represents between 21% to 33% of the links depending on the connectivity of the datasets. The set of hidden links is split into a validation set, used to control the convergence during the training phase, and a test set, used to calculate the precision and construct the ROC curves. We compare CRTM, with the context matching the sentence in which the link occurs, against *ad hoc* variants of RTM. gRTM was considered then discarded because of its prohibitive runtime on large corpora.

- **CRTM**: We define the context as the sentence in which the hyperlink occurs. After gridsearching we set the learning rate l to 0.01 in the update rule for Q, which yield the best results. Word embeddings are trained independently on each dataset using Skip-Gram with negative sampling (window size of 10 and 20 negative samples per true sample) [33]. By training one word-embedding set per corpora, we did not introduce external knowledge to our model and remain fair to the other baselines.
- **RTM**: We train the model following [10], with the original link function in Eq. 1. However this link function doesn't allow ranking the words because it is independent to the link's location. Hence, during the test phase, we change RTM's link function to $\psi_{(d,d')}(y=1;w_i)=\exp(\eta^\top(z_{d,i}\circ\overline{\mathbf{z}}_{d'})+\nu)$, *i.e.* we replace $\overline{\mathbf{z}}_{d'}$ with $z_{d,i}$, the topic assignment for word i. This trick allows us to measure the probability of each word w_i to carry the link. Note that we are limited to $z_{d,i}$, as considering larger context at test time would require modifications akin those proposed in CRTM.
- **TAGME**: In order to evaluate TAGME we use the annotation service provided by the D4Science's API[3]. For each hidden hyperlink we report if TAGME identify the right linked document and the corresponding anchor. As TAGME only output one result per linked document because of its disambiguation process, only P@1 is reported. We set the confidence score threshold at 0.1, discarding all results below this value. This value is the smaller threshold recommended by the documentation.

We also consider simpler variants of CRTM for the purpose of the ablation study:

- **CRTM$_1$** (no context): We consider singleton contexts that consist only contain the word carrying the link.
- **CRTM$_U$** (uniform weighting): We match the contexts with sentences and remove the attention mechanism by directly calculating $\overline{\mathbf{z}}_{d,c}$ as a simple average. Because this link function is doesn't take the exact location of the word into account, we again use the trick that consists in restricting the context to a single word at test time.
- **CRTM$_P$** (positional weighting): We match the contexts with sentences and replace the attention weights with weights calculated according to a Gaussian smoothing centered on the position of the word carrying the link. The weight for a word w_i in the context is calculated as $g_i=e^{-\frac{1}{2}\left(\frac{\mathrm{dist}(i)}{\sigma}\right)^2}$, where $\mathrm{dist}(i)$ is the distance between w_i and w_{link}, the word carrying the link. σ is the standard deviation. We set it to $\sigma=3$ by gridsearch.
- **CRTM$_I$** (no representation learning): We match the contexts with sentences and remove the additional parameters by setting $Q=I_K$, with I_K the identity matrix.

All models are trained with 50 topics, which we've found to give the best performance for [10] and CRTM.

[3]https://sobigdata.d4science.org/web/tagme/tagme-help

Table 1: Dataset properties.

Category	*Physics*	*Society*	*Fisica*	*Società*	*Physik*	*Gesellschaft*
Language	English	English	Italian	Italian	German	German
Nb of pages	6327	14486	2254	3106	3438	12262
Nb of links	18821	30749	7145	7093	10727	35710
Average number of words per doc	183	194	121	130	120	111
Average number of sentences per doc	9.77	10.05	5.79	5.94	8.69	8.12
Average degree	5.9	4.2	6.3	4.5	6.2	5.8

We set the hyperparameter α to 5.0 for all models which leads to consistently good performances for *CRTM* and RTM (identical to the value chosen by [10] in their evaluations), and set $\rho = 2000$ negative examples for the regularization in η and ν estimation.

4.3.2 Link Prediction. We hide 10% of edges before training models. For the four baselines learning documents embeddings, we compare the cosine similarity between hidden pairs and 10 negative examples of unconnected documents. For both RTM and CRTM we use the link function defined at Equation 1 and compare the probability of the true link and 10 negative examples. Because CRTM link function doesn't take into account all the content of the source document, Equation 2 is not suitable for link prediction. Switching from Equation 2 to 1 doesn't use the parameter Q learned by CRTM and this modification will be discussed in Section 4.5.

We compare CRTM to RTM and 4 recent document network embedding methods specifically designed for link prediction:

- **CRTM**: The context is still defined as the sentence in which the anchor link occurs and we use the same parameters and word-embeddings used for the previous evaluation. We use Equation 1 for computing the probability of two documents being linked.
- **RTM**: We use the parameters previously described for anchor prediction.
- **RLE**: The parameter λ is set to 0.7, and RLE outputs document embeddings in dimension $d = 160$. The word embeddings are obtained with Skip-Gram with negative sampling, using a window of 15 words and 5 negative examples per true sample, as described in [17].
- **GELD**: We fix $\alpha = 0.85$ after gridsearching, and run the model for 40 epochs. GELD outputs vectors in dimension $d = 160$. The word embeddings are obtained with Skip-Gram with negative sampling, using a window of 15 words and 5 negative examples per true sample.
- **TADW**: Following [44], we reduce the dimension of word vectors to 200 via SVD decomposition of the TFIDF matrix, select $k = 80$ and $\lambda = 0.2$. We run TADW for 20 epochs.
- **GVNRt**: We run 80 random walks per node of length 40. We fix the window size at 5. GVNRt run for 20 epochs and document embeddings are obtained by concatenating matrix I and J.

4.4 Anchor Prediction

Table 2 reports the average precision at 1 and the precision at 5 for all models over 10 runs, for all five datasets. Figure 1 shows averaged ROC curves for CRTM and RTM on each datasets.

4.4.1 Comparison of CRTM with RTM. CRTM outperforms RTM by a large margin on all datasets. In particular, CRTM exceeds 0.5 in precision at 1 on all English and German datasets, when RTM only score 0.41 in the most favorable case. This superiority of CRTM is also illustrated by the ROC curves on Figure 1. *CRTM* also outperforms RTM in the precision at 1 on the two Italian corpora, but performs equally as good as RTM in the precision at 5. This can be explained in part by the fact that, even though Italian and English documents have the same number of average number words per sentence (about 21.5 in Italian and 19.7 in English), the Italian corpora are smaller, and Italian documents are shorter but with a greater average degree. This may degrade CRTM's ability to generalize over anchor modelization. A similar explication could be given for the results in the German corpora. German and English datasets tends to have a similar average number of sentences per document, but German sentences are much shorter on our datasets. Furthermore the German language uses declensions and much more surcomposition than English in its vocabulary which could lead to more unique words, and so a weaker topic assignation of those words.

Overall, CRTM manages to rank the words carrying the hidden links higher than RTM, as evidenced by the ROC curves in Figure 1.

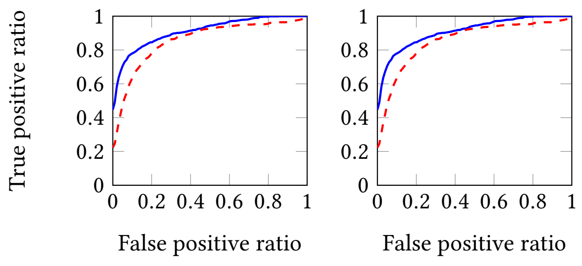
4.4.2 Comparison of CRTM and TAGME. . TAGME outperforms CRTM in terms of P@1 on the two German corpora, and on the *Physics* one. However CRTM exceeds its competitor on the *Society* by 0.02 in precision at 1. For the two Italian datasets TAGME still achieves better performance on P@1, but close to CRTM's one on the *Società* dataset. Yet those results may be tempered by the fundamental differences between CTRM and TAGME. As TAGME relies on a predefined dictionary made of previously observed anchors and documents titles, the annotation process is deterministic and TAGME can't infer unseen anchors. On the other hand CRTM aims to infer anchors depending on their topic and context. This means that top words predicted by CTRM are not necessarily inappropriate, even though they didn't match the exact anchor of the removed hyperlink. By saying this we could mitigate those results and advance that CRTM's P@5 performance, exceeding by a large margin TAGME's P@1 on all datasets, can lead to a better diversity of links. We further illustrate this point in Section 4.6.

Table 2: Average P@1 and P@5 per corpus for CRTM and baseline (standard deviation in parentheses).

Corpus	*Physics*		*Society*		*Fisica*		*Società*		*Physik*		*Gesellschaft*	
	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
CRTM	.65 (.02)	**.86** (.02)	**.65** (.03)	**.88** (.01)	.45 (.04)	**.76** (.03)	.38 (.02)	**.69** (.01)	.50 (.02)	**.81** (.001)	.53 (.02)	**.81** (.01)
RTM	.34 (.03)	.71 (.03)	.32 (.02)	.69 (.02)	.33 (.02)	.72 (.02)	.31 (.02)	.68 (.02)	.39 (.03)	.75 (.02)	.41 (.02)	.75 (.01)
TagMe	**.71** (.00)	-	.63 (.00)	-	**.51** (.01)	-	**.40** (.01)	-	**.65** (.01)	-	**.60** (.01)	-
CRTM$_1$	.37 (.02)	.78 (.02)	.36 (.02)	.76 (.01)	.33 (.03)	.73 (.02)	.33 (.02)	.67 (.03)	.39 (.01)	.74 (.01)	.41 (.02)	.76 (.01)
CRTM$_U$	.37 (.02)	.76 (.02)	.36 (.02)	.74 (.01)	.32 (.02)	.70 (.02)	.33 (.02)	**.69** (.01)	.39 (.02)	.76 (.02)	.42 (.01)	.76 (.01)
CRTM$_P$	.37 (.01)	.78 (.01)	.36 (.01)	.75 (.02)	.34 (.02)	.72 (.01)	.33 (.02)	.67 (.03)	.39 (.03)	.75 (.02)	.41 (.01)	76 (.01)
CRTM$_I$	.37 (.02)	.76 (.02)	.35 (.01)	.74 (.02)	.33 (.03)	.72 (.02)	.33 (.02)	.67 (.03)	.38 (.005)	.74 (.003)	.42 (.01)	.77 (.01)

Table 3: Average wall-clock runtime in seconds (time to train the embeddings in parentheses when applicable).

	Average runtime (s)	Average runtime/iter (s)
CRTM	$2.2 \times 10^2 (+10^2)$	28.4
RTM	4.6×10^2	11.4
CRTM$_1$	14.3×10^2	12.2
CRTM$_U$	6.1×10^2	18.8
CRTM$_P$	6.4×10^2	18.2
CRTM$_I$	$5.1 \times 10^2 (+10^2)$	12.73



(a) *Physics* (b) *Society*

Figure 1: Averaged ROC curves of CRTM (blue solid line) and RTM (red dashed line) on the two English corpora.

4.4.3 Execution time. We report in Table 3 the average wall-clock execution time for all models, across all datasets. All models are coded in Python 3 and trained using a single core on a computer with an *Intel i7-6700K* CPU and 64GiB of RAM. We notice that all the average runtimes are in $O(10^2)$ seconds, including the time required to train the word embeddings in *CRTM*. Interestingly, while an iteration takes longer to compute in *CRTM*, more than twice the time needed for RTM, CRTM needs fewer iterations to converge and thus ends up running faster than RTM.

4.4.4 Ablation Study. In this section, we study the performance of CRTM and its variants to show the importance of each of its components.

Impact of the attention mechanism. To judge the impact that the attention mechanism has on the performance of CRTM, we must compare it with these of CRTM$_1$ (no context), CRTM$_U$ (simple average) and CRTM$_P$ (Gaussian smoothing around the word's position). CRTM$_1$ has slightly better performance than RTM and is largely outperformed by CRTM. This tends to show that restricting the context to the sole word carrying the link is often too drastic, which can also explain why RTM struggle to match the performance of CRTM. CRTM$_U$ manages to equal CRTM on the *Società* dataset in precision at 5. However it is consistently outperformed by CRTM in all the other settings, and in particular in precision at 1. This suggests that simply averaging the per-word topic assignments is sub-optimal. On the other hand, CRTM$_P$ manages to improve over RTM, with a relative gain of up to 7% in the most favorable cases. This seems to indicate that the topic assignments of the words closest to the one carrying the hyperlink are more important. Yet, CRTM$_P$ is still outperformed by CRTM in the precision at 1 and 5 on all datasets, which shows that the attention mechanism is an important component of CRTM.

Impact of Q. Here we compare the performance of CRTM with the performance of CRTM$_I$ (where $Q = I_K$). CRTM consistently outperformed CRTM$_I$ in precision at 1 and 5 on the six considered corpora. We can interpret that by saying that learning new representations is beneficial.

4.5 Link Prediction

Table 4 reports the AUC scores for all datasets. We run each model five times, on each dataset, and report mean AUC with standard deviation.

Results analysis. CRTM beats TADW by a clear margin on five of six datasets. CRTM's performance matches RTM or slightly improves upon it, while coming close to RLE and GELD on English datasets. GVNR-t clearly performs better than all methods. Still, we observe that CRTM achieves consistent performances, with at least 0.80 AUC in all cases. It is worth noting that GVNR-t has a training time of several hours on our datasets (due to the computational costs of random walks), while CRTM is trained in less than 20 minutes on the same computer. This experiment shows that taking into account the context of anchor links at least doesn't degrade CRTM results in link prediction w.r.t RTM, and leads in some cases to a minor performance boost.

Impact of Q. As mention in section 4.3.2, CRTM's link function is not suitable for link prediction as it requires knowing the anchor. Therefore once we've trained CRTM we substitute its link function

Table 4: AUC for link prediction (standard deviation in parentheses)

	Physics	*Society*	*Fisica*	*Società*	*Physik*	*Gesellschaft*
CRTM	.87 (.004)	.87 (.003)	.81 (.01)	.78 (.02)	.82 (.002)	.80 (.01)
RTM	.85 (.02)	.86 (.01)	.80 (.01)	.78 (.03)	.82 (.005)	.78 (.01)
RLE	.89 (.002)	.91 (.003)	.90 (.002)	.89 (.001)	.86 (.002)	.88 (.001)
GELD	.89 (.001)	.91 (.001)	.89 (.001)	.88 (.002)	.87 (.001)	.85 (.001)
TADW	.83 (.003)	.82 (.003)	.80 (.005)	.73 (.001)	.67 (.003)	.65 (.004)
GVNR-t	.97 (.03)	.96 (.04)	.95 (.001)	.95 (.04)	.96 (.03)	.96 (.04)

with the simpler RTM's link function to do link prediction, thus omitting the transformation Q. We further investigate the impact of this parameter by introducing the transformation Q in the link function of RTM.

The overall average AUC in link prediction on the six datasets falls to 0.63, while it is about 0.80 without using the parameter Q. This result, coupled with Section 4.4.4's conclusion, suggest that (i) CRTM learns topics that are useful for link prediction and that (ii) Q confers it the ability to efficiently recombine topics so that they are suited to anchor prediction. This means that a single CRTM model could solve both the link prediction (even though it doesn't match the performance of the most up-to-date techniques) and anchor prediction tasks, by simply changing the link function at prediction time.

Table 5: Five most likely anchors in the "Semiconductor" page, connected to the "Transistor" page.

Transistor, A transistor is a semiconductor device used to amplify or switch electronic signals and electrical power. [...]

↑

Semiconductor

CRTM	**RTM**
transistor	loom
MOSFET	MOSFET
semiconductor	transistor
circuit	enable
Moore	circuit

Table 6: Five most likely anchors in the "Computer" page, connected to the "Transistor" page.

Transistor, A transistor is a semiconductor device used to amplify or switch electronic signals and electrical power. [...]

↑

Computer

CRTM	**RTM**
transistor	staircase
electron	atom
MOSFET	dopant
dopant	fabricated
electrical	ability

4.6 Case Study

Here we show the anchors that CRTM is able to automatically detect, given a pair of source and target documents. To this end, CRTM was trained with all hyperlinks removed in the source documents, to prevent it from simply listing anchors seen during training. These anchors could be used by writers, to automatically insert hyperlinks towards related pages. They could also be useful for readers, as they could serve as contextual hyperlinks, linking the page they are reading with those they've previously read. We also report the anchors found by RTM, to highlight how those found by CRTM are more relevant.

As an example, Table 5 shows the five most likely anchors, given the page about "Semiconductor" as the source, and the page about "Transistor" as the target, and Table 6 shows the five most likely anchors, given the page about "Computer" as the source, and the page about "Transistor" as the target. We note that CRTM always rank the word transistor first, the most natural word to be an anchor. RTM actually manages to identify it as an important word too, but ranks it lower. CRTM also ranks MOSFET at the second or third place, which is a type of transistor. In addition, we observe that RTM predicts some less interesting words, such as *loom*, *fabricated* or *enable*, while CRTM highlights relevant but less obvious terms, like *Moore* (from the Moore's law).

5 CONCLUSION AND FUTURE WORK

We have presented the Contextualized Relational Topic Model, CRTM, a probabilistic modeling framework to infer latent topics in networks of documents, that explicitly accounts for the locations of the links in the text. We've experimentally shown the relevancy of our approach through a quantitative evaluation based on several Wikipedia datasets, in English, Italian and German. We've also shown that CRTM has a competitive runtime, which makes it usable in practice to solve tasks akin to anchor prediction without relying on external information like a knowledge graph. We also demonstrated that taking anchor links into account doesn't degrade the model's performance in link prediction. From a qualitative point of view, we've illustrated how CRTM can assist knowledge bases contributors while they specify anchor links after writing. In future work, we'd like to investigate more complex link functions based upon a more sophisticated modeling of topics and documents, and extend our work to more recent works, like Graph Neural Networks and Neural Topic Models.

REFERENCES

[1] Christopher Aicher, Abigail Z Jacobs, and Aaron Clauset. 2013. Adapting the stochastic block model to edge-weighted networks. *arXiv preprint arXiv:1305.5782* (2013).

[2] Haoli Bai, Zhuangbin Chen, Michael R Lyu, Irwin King, and Zenglin Xu. 2018. Neural relational topic models for scientific article analysis. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 27–36.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.

[4] Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer.

[5] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[6] Aleksandar Bojchevski and Stephan Günnemann. [n.d.]. Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking. In *Proceeding of ICLR*.

[7] Robin Brochier and Frédéric Béchet. 2021. Predicting Links on Wikipedia with Anchor Text Information. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[8] Robin Brochier, Adrien Guille, and Julien Velcin. 2019. Global vectors for node representations. In *The World Wide Web Conference*. 2587–2593.

[9] Robin Brochier, Adrien Guille, and Julien Velcin. 2020. Inductive Document Network Embedding with Topic-Word Attention. In *Proceedings of ECIR*. Springer, 326–340.

[10] Jonathan Chang and David Blei. 2009. Relational topic models for document networks. In *Artificial Intelligence and Statistics*. 81–88.

[11] Jonathan Chang and David M Blei. 2010. Hierarchical relational models for document networks. *The Annals of Applied Statistics* (2010), 124–150.

[12] Ning Chen, Jun Zhu, Fei Xia, and Bo Zhang. 2013. Generalized relational topic models with data augmentation. In *Proceedings of IJCAI*. 1273–1279.

[13] Ran Ding, Ramesh Nallapati, and Bing Xiang. 2018. Coherence-Aware Neural Topic Modeling. In *EMNLP*.

[14] 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*. 1625–1628.

[15] Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep Joint Entity Disambiguation with Local Neural Attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2619–2629.

[16] Martin Gerlach, Marshall Miller, Rita Ho, Kosta Harlan, and Djellel Difallah. 2021. Multilingual Entity Linking System for Wikipedia with a Machine-in-the-Loop Approach. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3818–3827.

[17] Antoine Gourru, Adrien Guille, Julien Velcin, and Julien Jacques. 2020. Document Network Projection in Pretrained Word Embedding Space. In *ECIR*. Springer, 150–157.

[18] Amit Gruber, Michal Rosen-Zvi, and Yair Weiss. 2008. Latent topic models for hypertext. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*. 230–239.

[19] Amit Gruber, Michal Rosen-Zvi, and Yair Weiss. 2008. Topic Models for Hypertext: How Many Words is a Single Link Worth? (2008).

[20] Amit Gruber, Yair Weiss, and Michal Rosen-Zvi. 2007. Hidden topic markov models. In *Artificial intelligence and statistics*. PMLR, 163–170.

[21] Xianpei Han and Le Sun. 2012. An entity-topic model for entity linking. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 105–115.

[22] Yu Hao, Xin Cao, Yixiang Fang, Xike Xie, and Sibo Wang. 2020. Inductive Link Prediction for Nodes Having Only Attribute Information. In *Proceedings of IJCAI*. 1209–1215.

[23] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. 782–792.

[24] Thomas Hofmann. 1999. Learning the Similarity of Documents: An Information-Geometric Approach to Document Retrieval and Categorization. In *Proceedings of NeurIPS*. 914–920.

[25] Diederik P Kingma, Max Welling, et al. 2019. An Introduction to Variational Autoencoders. *Foundations and Trends® in Machine Learning* 12, 4 (2019), 307–392.

[26] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.

[27] Martin Josifoski Sebastian Riedel Luke Zettlemoyer Ledell Wu, Fabio Petroni. 2020. Zero-shot Entity Linking with Dense Entity Retrieval. In *EMNLP*.

[28] Jie Liu, Zhicheng He, Lai Wei, and Yalou Huang. 2018. Content to node: Self-translation network embedding. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. 1794–1802.

[29] Víctor Martínez, Fernando Berzal, and Juan-Carlos Cubero. 2016. A Survey of Link Prediction in Complex Networks. 49, 4 (2016).

[30] Olena Medelyan, Ian H Witten, and David Milne. 2008. Topic indexing with Wikipedia. In *Proceedings of the AAAI WikiAI workshop*, Vol. 1. 19–24.

[31] Qiaozhu Mei, Deng Cai, Duo Zhang, and ChengXiang Zhai. 2008. Topic modeling with network regularization. In *Proceedings of the 17th international conference on World Wide Web*. 101–110.

[32] 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*. 233–242.

[33] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[34] Feng Nan, Ran Ding, Ramesh Nallapati, and Bing Xiang. 2019. Topic Modeling with Wasserstein Autoencoders. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 6345–6381.

[35] Francesco Piccinno and Paolo Ferragina. 2014. From TagME to WAT: a new entity annotator. In *Proceedings of the first international workshop on Entity recognition & disambiguation*. 55–62.

[36] C Estelle Smith, Bowen Yu, Anjali Srivastava, Aaron Halfaker, Loren Terveen, and Haiyi Zhu. 2020. Keeping community in the loop: Understanding wikipedia stakeholder values for machine learning-based systems. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–14.

[37] Congkai Sun, Bin Gao, Zhenfu Cao, and Hang Li. 2008. HTM: A topic model for hypertexts. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. 514–522.

[38] Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015. Modeling mention, context and entity with neural networks for entity disambiguation. In *Twenty-fourth international joint conference on artificial intelligence*.

[39] I Tolstikhin, O Bousquet, S Gelly, and B Schölkopf. 2018. Wasserstein Auto-Encoders. In *6th International Conference on Learning Representations (ICLR 2018)*. OpenReview. net.

[40] 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*. 5998–6008.

[41] Hao Wang, Xingjian Shi, and Dit-Yan Yeung. 2017. Relational deep learning: A deep latent variable model for link prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31.

[42] Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1006–1011.

[43] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2013. A biterm topic model for short texts. In *Proceedings of the 22nd international conference on World Wide Web*. 1445–1456.

[44] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y Chang. 2015. Network representation learning with rich text information. In *Proceedings of IJCAI*. 2111–2117.

[45] Weiwei Yang, Jordan Boyd-Graber, and Philip Resnik. 2015. Birds of a feather linked together: A discriminative topic model using link-based priors. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 261–266.

[46] Weiwei Yang, Jordan Boyd-Graber, and Philip Resnik. 2016. A discriminative topic model using document network structure. In *Proceedings of the 54th ACL Annual Meeting (Volume 1: Long Papers)*. 686–696.

[47] Aonan Zhang, Jun Zhu, and Bo Zhang. 2013. Sparse relational topic models for document networks. In *Joint ECML and KDD*. Springer, 670–685.

[48] Jian Zhang, Jun Zheng, Jinyin Chen, and Qi Xuan. 2020. Hyper-Substructure Enhanced Link Predictor. In *Proceedings of the 29th ACM International Conference on Information Knowledge Management*. Association for Computing Machinery, New York, NY, USA, 2305–2308. https://doi.org/10.1145/3340531.3412096

[49] Deyu Zhou, Xuemeng Hu, and Rui Wang. 2020. Neural Topic Modeling by Incorporating Document Relationship Graph. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 3790–3796.

[50] Qile Zhu, Zheng Feng, and Xiaolin Li. 2018. GraphBTM: Graph enhanced autoencoded variational inference for biterm topic model. In *Proceedings of the 2018 conference on empirical methods in natural language processing*. 4663–4672.