Content Explorer: Recommending Novel Entities for a Document Writer

Michal Lukasik
Google AI
mlukasik@google.com

Richard Zens
Google AI
zens@google.com

Abstract

Background research is an essential part of document writing. Search engines are great for retrieving information once we know what to look for. However, the bigger challenge is often identifying topics for further research. Automated tools could help significantly in this discovery process and increase the productivity of the writer. In this paper, we formulate the problem of recommending topics to a writer. We consider this as a supervised learning problem and run a user study to validate this approach. We propose an evaluation metric and perform an empirical comparison of state-of-the-art models for extreme multi-label classification on a large data set. We demonstrate how a simple modification of the cross-entropy loss function leads to improved results of the deep learning models.

1 Introduction

An important part of writing about some topic is researching relevant background material, which can be a challenging and tedious task. If the author has a clear idea what to write about, web search engines provide an excellent tool for retrieving information. However, the author often first has to spend time discovering which topics are related, i.e., she needs to conduct *exploratory search* (White and Roth, 2009). This is challenging with traditional keyword-based search engines like *Google*, which are tailored to providing the information that a user is explicitly searching for (Marie, 2014).

Exploratory search poses a challenge in the writing process. It has been shown that the *exploration* phase elicits strong negative feelings in students writing essays (Smith and Campbell, 1999; Kuhlthau, 1990) and that students consider the research activity to be at the cost of other pending commitments (Smith and Campbell, 1999). Moreover, exploratory search is more cognitively demanding than lookup search tasks (Marie, 2014). Automatic tools tailored to exploratory search for new content could significantly alleviate the difficulty the writer faces in the research phase.

Intuitively, the tool should suggest topics for further research which are relevant and interesting. Hence the suggestions need to be related to the document and should not be too obvious. For instance, a student writing about the *monarch butterfly* might benefit from a suggestion *milkweed* as it is both relevant and interesting. On the other hand, *insect* would be a rather poor suggestion as it is too obvious. To summarize, the key challenge we set out to address is: *For a given piece of text, what are the related topics and how important are they?* In Section 3.2 we describe a user study where annotators evaluated usefulness of entities, which we then use to identify characteristics of good suggestions.

It is worth mentioning that systems for providing automatic recommendations of entities to a document writer have recently been introduced by the industry, including Google Explore in Docs[1] and Microsoft Researcher[2]. The systems are proprietary and their design is not published, making the comparison not possible. To the best of our knowledge this is the first documented attempt to address the problem of recommending novel entities to a document writer. We hope this paper will attract the interest of the research community and inspire future work.

The contributions of this paper are:

1. Formulating the problem of recommending future entities to a document writer.

[1] http://bit.ly/2f6CVeR
[2] http://bit.ly/29XaPAJ

2. Conducting a user study to identify what topic suggestions users find useful.

3. Formulating the problem and defining an automatic evaluation metric, both motivated by the user study.

4. Evaluating state of the art approaches to extreme multi-label classification.

5. Demonstrating how a modified loss function helps improve over state of the art neural network models.

2 Related Work

Recommending topics to a document writer can be viewed in the context of different fields which we discuss below.

Exploratory Search White and Roth (2009) define exploratory search as "information seeking problem context that is open-ended, persistent, and multi-faceted" and "information-seeking processes that are opportunistic, iterative, and multi-tactical". Research on exploratory search focuses on supporting a user in the interactive and iterative process of seeking for information and includes: designing better interfaces, visualization of search results, clustering of results, supporting serendipitous discoveries, supporting different user profiles (Marie, 2014). Instead, in this work we aim to fully automate discovery of *relevant* and *interesting* topics to the document writer. Moreover, the input in our case is an initial portion of a document written by a user rather than a query or a single entity, as is usually the case in the information retrieval setups.

Extreme Multi-label Classification Extreme multi-label classification (XML) is an instance of a multi-label classification problem (i.e., where multiple labels can be assigned to a single example at the same time) under a large label space. There are multiple works investigating XML, including random forest (RF) (Prabhu and Varma, 2014; Jain et al., 2016) and embedding (Bhatia et al., 2015) approaches. These methods rely on bag of words feature representation, ignoring the sequential nature of text. Neural networks (NNs) have been successful in modeling NLP tasks through their ability to learn structure, however have not been widely studied for XML problems. Covington et al. (2016) applied NNs to YouTube video recommendation, and Liu et al. (2017) showed a convolutional neural network architecture to outperform strong baselines on a range of NLP tasks. The importance of label weighting for recommending rare items has not been considered in the NNs for extreme multi-label classification, which is a central problem in our task, as in other tasks with extreme label spaces (Jain et al., 2016). In Section 5 we describe the state of the art RF and NN approaches to XML.

Entity Retrieval and Tagging Assigning entities to an input has been considered, however in different contexts. In tag recommendation, a text is summarized with a set of entities (Song et al., 2011) and in entity search one needs to answer queries about entities against a set of documents and entities (Cheng et al., 2007). These tasks are different as no new content is predicted.

The Related Entity Finding (REF) challenge of the Text Retrieval Conference (TREC) considered a task where given a source entity, a relation and a target type, a target entity needs to be identified satisfying the required relation (Balog et al., 2010). This is a different problem since no specification for relations between the entities is provided in the input document from a user.

The work on semantic relatedness of entities (Milne and Witten, 2008) is different since, as explained in the introduction, good entity recommendations are not necessarily those which are semantically related to entities from the user text. Bordino et al. (2013) considered retrieving entities related to a query in the question answering scenario. The authors did not set the problem in a supervised learning setup and instead find entities closest in terms of similarity of documents containing them. In contrast, in Section 3.2 we justify a supervised learning setup of the task.

The TREC Complex Answer Retrieval Track (CAR)[3] is a challenge where based on a document outline, related text passages and entities are retrieved (Dietz et al., 2017). In this formulation it is assumed that a general outline of what a document author intends to cover is given, thus providing a clear guidance for what is relevant. Instead, our input is an initial part of a document and we seek to find novel entities based on the input.

[3] http://trec-car.cs.unh.edu

Recommending Rare Items Information retrieval applications emphasized the importance of retrieving rare labels rather than common ones which are likely to be already known to a user. Baeza-Yates and Ribeiro-Neto (1999) defined novelty of a set of recommendations as the proportion of unknown items to the user, a challenging definition to work with when user's knowledge is unknown (Hurley and Zhang, 2011). Bordino et al. (2013) explored the problem of retrieving *serendipitous* results when retrieving answers to queries. The authors built an information retrieval system based on finding entities most often co-occurring with a query entity and employed IDF (inverse document frequency) for filtering out overly generic answers. Also many other works employ IDF for rewarding rare items (Zhou et al., 2010; Vargas and Castells, 2011; Wu et al., 2014; Jain et al., 2016). Here we take a supervised learning approach to the entity recommendation problem, demonstrate the usefulness of IDF scoring in the context of our problem with a user study, and utilize IDF in the evaluation metric.

Hurley and Zhang (2011) define novelty of an item in the context of the set of relevant recommendations using an average dissimilarity from other items in the set. Similarly, the Maximal Marginal Relevance metric evaluates a set of retrieved items in an information retrieval problem by rewarding the diversity of the set (Carbonell and Goldstein, 1998). Note this is a different notion of novelty from that considered in this work.

3 Investigating the Problem

In this section we investigate the problem formulation and what it means for an entity to be a useful suggestion for a user.

3.1 Problem Definition

Let a document d be represented as a sequence of entities $(e^1, \ldots, e^{|d|}) = E$. We partition these entities into two sets C_d and F_d: those which occur in the first h sentences (in our case $h = 10$) and those which don't. Hence, $C_d \cap F_d = \emptyset$ and $C_d \cup F_d = E$.

We are not interested in retrieving all entities that could occur in the future, because it is not feasible to provide all such recommendations to a user. Instead, we focus on ranking the target entities, and selecting the most relevant k to suggest, which is a standard practice in retrieval tasks (Jain et al., 2016).

3.2 What Entities are Good Suggestions?

In this section we describe a user study we ran for testing the hypotheses about which entities constitute useful recommendations. In particular, we test whether what a user writes next in a document is actually considered to be a good recommendation by the raters and whether IDF score correlates with how useful an entity is considered.

Human Evaluation Dataset We consider 1000 Wikipedia documents for a human evaluation study. For each document, let E denote the set of all entities that occur in the document and E_p denote entities occurring in the initial passage p. We found 5 most co-occurring entities from $E \setminus E_p$[4] with the entities from E_p across sentences from a large scale web documents corpus (the dataset described later in Section 4). Annotators were then asked to rate entity suggestions against an input passage p in terms of how *useful* they are in the process of continuing to write the document. Each entity was rated by 3 annotators with a score ranging from 1 to 5 and the total number of annotators was 845. To evaluate agreement among raters, we employ the Intraclass Coefficient for consistency[5] (ICC; (McGraw and Wong, 1996)) for two-way random effects model with the effects corresponding to a rater and a rated entity. We found the average score ICC to be equal $\text{ICC}(\text{C}, 3) = 0.69$. We refer the reader to the supplemental material for details about how ICC was applied.

Evaluating the Supervised Setup We found that the mean rating for entities which actually occur in the future is 3.24 ± 0.83, whereas the mean rating for the other entities is 2.59 ± 0.93. Running an independent T-test for comparing means between the two groups yields a p-value $p < 0.001$. This supports the supervised learning setup of entity suggestion, where we split documents from a corpus into two parts, and unseen entities from later parts form labels for the initial parts.

Does IDF Correlate with Usefulness? One of the requirements we set for the entity recommendation is the interestingness of the entities. As

[4] I.e., excluding entities occurring in the passage.

[5] Intraclass Coefficient is an approach to evaluate inter-rater reliability when ratings are organized into groups, as in our case, where entities are grouped into passages against which they are scored.

reported in Section 2, one popular approach to recommending rare items in information retrieval tasks is weighting their utility by their IDF score (Bordino et al., 2013; Jain et al., 2016). Let us analyze in the document writing setup how useful users perceive entities which score high in IDF. To this end, we measure how the *ground truth reviewer* (the average rating given by the human annotators to an item) correlates against the baseline rating *B* and how it correlates against the IDF weighted rating *W*. Here, the baseline rating *B* simply rates an entity with 0 if it does not occur in the future of the document and with 1 if it does occur in the future of the document. The IDF weighted rating *W* rates an entity with its IDF score multiplied by the score returned by the reviewer *B*. We employ ICC(A, 1) to find the agreements between pairs of ratings. We find ICC(A, 1) between the *ground truth reviewer* and *B* to be 0.30, and between the *ground truth reviewer* and *W* to be 0.42. This shows that entities with high IDF scores are perceived as more useful. Note the potential space for improvement by finding a metric which would yield even higher correlation against the human rating than IDF.

3.3 Automatic Evaluation

Recall that we aim at rewarding entities which are both *relevant* (i.e., within the target set of entities) and *interesting* (i.e., are not obvious, as the entity *insect* was for the *monarch butterfly* example from Section 1). Since we care only about the k highest ranked recommendations from the system, a potentially useful metric for evaluation is prec@k$=\frac{1}{k}\sum_{j=1}^{k} I_{\{p^j \in F_d\}}$, where p^j are the predictions. Even though prec@k captures relevancy, it fails to distinguish between generic and specific entities. Documents tend to contain many entities in the target set, some of which are generic (e.g. *insect*). Prec@k scores all entities the same and rewards predictions of generic entities occurring across almost all web documents (e.g. *Internet*), which are arguably not interesting.

We propose a metric based on cumulative gain (Järvelin and Kekäläinen, 2002), where we use IDF for scoring the relevance of labels:

$$\text{CG-IDF@k} = \sum_{j=1}^{k} I_{\{p^j \in F_d\}} \text{IDF}(p^j).$$

To facilitate interpretability, we use a normalized version of CG-IDF@k, the normalized cumulative gain (NCG-IDF@k). NCG-IDF@k is obtained by dividing CG-IDF@k by the maximum sum of IDF scores of k entities from the target set.

Why would IDF help make entities more interesting? As shown by the user study in Section 3.2 IDF correlates with how relevant an entity is perceived by a user. Moreover, as reported in Section 2, IDF is widely used for boosting rare, novel items (Bordino et al., 2013; Jain et al., 2016).

4 Dataset

We consider the problem of recommending topics that an author writing a document might be interested in writing about next. In Section 3.2 we used a human evaluation study to support a supervised learning setup for this problem, where what is written later in a document is deemed to be a good recommendation for an initial part. The Web is a rich source of documents which facilitates construction of large datasets. Below we describe details of how we construct the dataset composed of web documents and report basic statistics thereof.

Construction We collected a dataset of 10M web documents with high pagerank (Page et al., 1998) scores across the Web as of November 2017. We ran entity recognition and linking to the Freebase knowledge graph using the Google NLP cloud.[6] Moreover, only the 100K most frequent entities are kept.[7] We randomly select 10K documents for the test set and 10K for the validation set which we use for hyperparameter tuning.

Statistics As shown in Figure 1 the document frequencies for the 100K most frequent entities from the dataset follow a power law distribution, with a small number of very frequent entities and many infrequent ones. The average number of documents per target entity is 4694.95, which is 3 orders of magnitude smaller than the maximum frequency. We found that the average number of future entities per document is 96, and the average number of input entities is 10. In Figure 2 we show the percentage of times a context entity is followed by a particular target entity (row and column, respectively). Notice the matrix is asymmetric. For

[6] http://tinyurl.com/h246dnz

[7] 100K entities are kept due to the challenges in handling larger output spaces by the models. Scaling to larger output spaces is possible but requires modifications which we leave for future work.

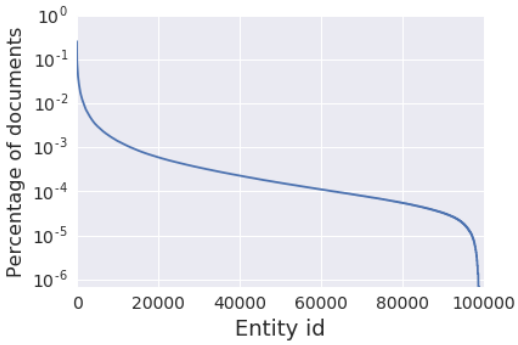


Figure 1: Percentage of documents for which each target entity occurs.

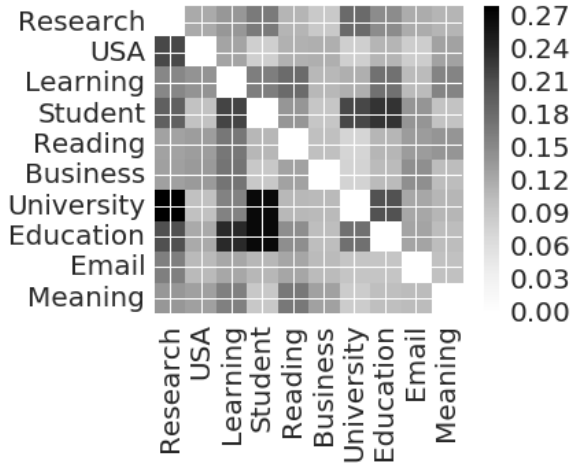


Figure 2: A heatmap showing what percentage of times a context entity (row) is followed by a particular target entity (column).

instance, USA is more likely to be followed by Research than the other way around.

5 Baselines and Models

In this section we describe the models that we applied to the entity recommendation problem.

5.1 Linear Models

Let us represent input entities via n-hot vector representation, i.e., vector entries corresponding to entities present in the input set are set to one, whereas other entries are set to zero. We consider *linear models*, where such an input vector is multiplied by a square weight matrix of size #entities $\times$ #entities, and a resulting vector contains scores for predicted entities. There are different possibilities for filling the entries of the weight matrix. We considered multiple options:

1. $\mathrm{N}(C, F)$ obtained by putting the raw co-occurrences of the context entity C (corresponding to a row) and the future entity F (corresponding to a column).

2. $\mathrm{P}(F|C)$ obtained by normalizing the $\mathrm{N}(C, F)$ matrix row-wise.

3. $\mathrm{PPMI}(C, F)$ (positive pointwise mutual information) aims to show how much more likely an entity F is to occur for a context C compared to observing them independently (Jurafsky and Martin, 2000). PPMI is a popular preprocessing step on the co-occurrence matrix before applying dimensionality reduction, such as SVD (see Section 5.2) (Herbelot and Vecchi, 2015).

5.2 Matrix Factorization

Matrix factorization methods (MFM) are among the most popular approaches for recommendation systems (Koren et al., 2009). This approach works similar to the *linear model*, except that the resulting matrix is decomposed into a sequence of smaller matrices, the product of which approximates the *linear model*. This approach reduces the number of parameters, which speeds predictions, saves memory, and may improve robustness to overfitting. We employ SVD and reduce the rank of the resulting matrix of the *linear model* to 100. MFM is applied analogously to the *linear model*, namely a vector encoding input entities is mapped into a vector with scores of target entities.

5.3 Random Forests

FastXML FastXML (Prabhu and Varma, 2014) is one of the most popular approaches to extreme multi-label classification. Each tree in this random forest model is grown recursively by splitting each node by a separating hyperplane. The hyperplane weight vector is chosen by optimizing for nDCG score (a non-differentiable loss which poses issues in the NN framework) and is additionally regularized with ℓ_1 norm to induce sparsity. Each leaf node contains a probability distribution over labels. At prediction time, the distributions from reached leaf nodes across the trees are aggregated and a final ranking of entities is created.

PFastreXML PFastreXML (Jain et al., 2016) builds on the FastXML model by introducing two modifications. First, the nDCG loss function is

replaced by a propensity weighted nDCG (in our case, the IDF weighted nDCG), resulting in rare items ranked higher. Second, the final ranking is re-ranked using tail label classifiers, which aims at further improvements in how highly rare items are scored.

Experimental Setup The scale of our dataset is bigger than that reported by Jain et al. (2016) and consequently the size of the generated random forests tends to be very large. Therefore, apart from keeping most of the hyperparameters as reported by Jain et al. (2016), in order to limit the size of generated trees, we modify two hyperparameters: the maximum number of labels per leaf node is set to 50 (instead of 10), and the maximum number of training examples per leaf node is set to 50 (instead of 10). Even under such a setup, the size of the generated models is around 150GB.

5.4 Neural models

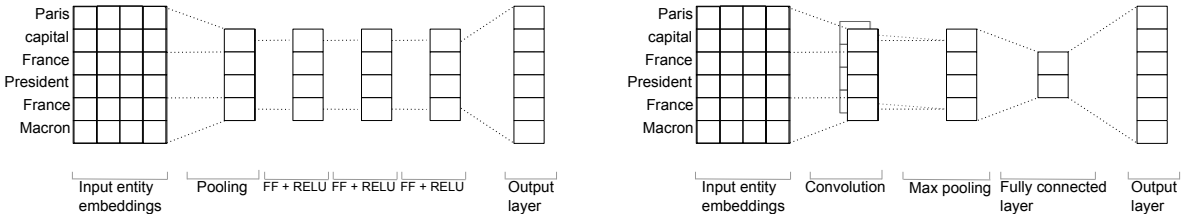
Recently NN models have been applied to extreme classification. In this section we discuss the YouTube neural model used for recommending YouTube videos (Covington et al., 2016) and next the XML-CNN model which has been demonstrated to achieve competitive results on a few NLP tasks (Liu et al., 2017).

The YouTube Model A NN model (depicted in Figure 3(a)) has been successfully applied for the YouTube recommendation problem (Covington et al., 2016). In the model, the input entities are embedded into a latent dimensionality V. The embeddings are then summed, and the resulting vector is passed through a number of layers, where in each layer an input vector is passed through matrix multiplication and a rectified linear unit (RELU) (Nair and Hinton, 2010). Afterwards, the resulting vector is converted into the space of dimensionality equal to the number of entities via an output layer. This way, scores are obtained over the entity space, which are used to choose the highest scored entities for predictions. The loss function is the cross-entropy (CE) between the soft-maxed activations and a uniform distribution over the target entities.

XML-CNN Convolutional neural networks (CNNs) have been successfully applied to a range of NLP problems (Kim, 2014; Bitvai and Cohn, 2015). Recently Liu et al. (2017) demonstrated how CNNs can be effective for XML problems. We depict their architecture in Figure 3(b). The input sequence is embedded in V dimensions, and passed through a convolutional layer, a fully connected layer and an output layer. After the convolutional layer the authors employ dynamic pooling which helps retain information about where in the input sequence the convolutions got triggered. The loss function is binary cross entropy (BCE), which considers labels individually rather than jointly. BCE is given by the formula $\text{BCE}(\mathbf{p}, \mathbf{y}) = \sum_{j=1}^{j=M} y_j \log(\sigma(p_j)) + (1 - y_j) \log(1 - \sigma(p_j))$, where σ is the sigmoid function. Lastly, a hidden bottleneck layer is used, which is motivated by introducing better generalization. In the experiments we did not find the dynamic max pooling to be beneficial, and instead we found max pooling with a higher number of filters (512 compared to 32 in (Liu et al., 2017)) to be better, keeping the number of parameters.

Modified Loss Function A potential problem with CE and BCE is that they reward all entities equally. However, as demonstrated in Section 3.2, some entities are more valuable than others. Since we would like to promote interesting entities over generic ones, we consider an alternative training loss function to BCE and CE which incorporates the IDF scores of target entities. We use CE, but instead of comparing the soft-maxed activations to a uniform distribution over target entities, we compare against normalized IDF scores from the training set. We call this loss function CE-IDF. When using CE-IDF with XML-CNN, we found that adding ℓ_2 norm of the weights and the cross-entropy regularization (Pereyra et al., 2017) helps prevent the model from overfitting. We select the hyperparameters controlling these two regularization terms using a held out validation set. This contribution is analogous to that of PFastreXML over FastXML due to Jain et al. (2016), where weighting the loss function in random forests by label propensity scores helps achieve better propensity weighted results.

Experimental Setup To regularize the networks we use a 50% dropout rate (Srivastava et al., 2014). We set the dimensionality of the embeddings to 1000, and the hidden layer size to 512. The hyperparameters for XML-CNN are set as reported by Liu et al. (2017), except for the ℓ_2 and cross-entropy regularization hyperparameters



(a) YouTube model (Covington et al., 2016)

(b) XML-CNN (Liu et al., 2017)

Figure 3: Neural network architectures applied to the entity recommendation problem.

	NCG-IDF				prec				model
	@1	@3	@5	@7	@1	@3	@5	@7	size
$p(F)$	14.86	14.50	14.61	14.56	36.76	32.69	31.12	30.23	28B
N(C, F) base	20.12	19.95	19.43	18.91	46.92	42.89	40.29	38.15	4.7GB
SVD	19.99	19.79	19.20	18.72	46.68	42.60	39.88	37.81	0.1GB
P$(F\|C)$ base	22.92	22.77	22.17	21.86	51.50	47.81	44.97	43.19	4.7GB
SVD	21.56	21.00	20.65	20.31	48.79	44.44	42.22	40.54	0.1GB
PPMI(C, F) base	22.92	20.86	19.45	18.42	23.95	21.17	19.33	18.01	4.7GB
SVD	18.22	18.22	18.03	17.89	26.49	25.36	24.31	23.49	0.1GB
Youtube base	31.30	31.12	31.07	31.02	54.85	52.62	51.09	49.79	1.8GB
IDF	32.95	32.24	32.00	31.62	50.95	49.07	47.63	46.57	1.8GB
XML-CNN base	33.13	32.20	31.93	31.75	58.72	53.60	52.05	50.35	1.8GB
IDF	33.89	33.42	33.02	32.76	51.99	49.54	48.40	47.24	1.8GB
FastXML	35.31	34.39	33.74	33.15	69.98	63.78	60.16	57.37	150GB
PFastreXML	36.19	34.94	34.12	33.31	55.09	51.51	49.50	47.85	150GB

Table 1: Experimental results for NCG-IDF@k and prec@k scores for different methods.

which were selected on the validation set. Note we optimize all parameters, including the entity embeddings, on the training data.

6 Experiments

In Table 1 we report results from the experiments on the 10M web documents dataset for prec and NDCG-IDF metrics for $k = 1, 3, 5, 7$, limiting k to small values as is common in the recommendation problems from large sets of items (Jain et al., 2016). The $p(F)$ baseline always predicts entities according to their frequency over the training set. It can be viewed as maximum likelihood estimate (MLE) for the model which is only composed of a bias vector (i.e., input entities are ignored). Notice the relatively high performance when the most popular entities are taken. For example, in 36.76% of cases entity *Research* (the most popular future entity from the corpus) is in the future of the document (as can be also seen from Figure 1, where the entity *Research* corresponds to the leftmost point of the graph). This constitutes a high value, as the vocabulary consists of $100K$ entities.

Among the *linear models*, P$(F|C)$ yields the highest scores, significantly outperforming the baselines. PPMI$(F|C)$ model yields relatively high NCG-IDF scores (although in most cases lower than P$(F|C)$), and very low precision scores. Notice that the SVD methods are consistently worse than the *linear models*. This shows that no additional generalization is gained when lowering the number of parameters of the *linear models*. When experimenting with higher ranks for SVD decomposition we found the performance increases, but does not improve over the *linear models*.

NNs improve over *linear models* according to both NCG-IDF and prec scores. This is especially apparent for NCG-IDF, where the relative improvement is very significant. XML-CNN is in all cases better than the Youtube model, which shows how utilizing more linguistic structure than simply bag of entities is helpful in the NN framework. Both Youtube and XML-CNN models with a modified loss function improve over the basic NN models in terms of the NCG-IDF metrics, showing that a simple adjustment of a loss function in the NN framework can lead to more rare

input	predictions			
	XML-CNN base	XML-CNN IDF	FastXML	PFastreXML
NASA recently released a study suggesting that the Antarctic Ice Sheet is gaining more ice than it is losing – a finding that at first blush seems to contradict the idea of global warming.	*glacier*, *Greenland*, *Earth*, *research*	*sea level rise*, *glacier*, *temperature*, *meltwater*	*ocean*, *research*, *temperature*, *understanding*	*glacier mass balance*, *ice shelf*, *glaciology*, *Greenland ice sheet*
Making your mobile web app talk: software Architecture conference. Microservices training O'reilly.	*learning*, *project*, *experience*, *information*	*application software*, *Javascript*, *presentation*, *organization*	*technology*, *service*, *learning*, *industry*	*technology*, *application software*, *project*, *Open Source*
Face recognition algorithms use a large dataset of photos labeled as having a face or not to estimate a function that predicts the presence y of a face from pixels x. As empirical economists, how can we use them?	*number*, *research*, *information*, *result*	*analysis*, *result*, *sample*, *statistics*	*information*, *data*, *number*, *result*	*result*, *analysis*, *set*, *R programming language*
Here we report the isolation of an arsenate reductase gene (PvACR2) from gametophytes that can suppress the arsenate sensitivity and arsenic hyperaccumulation phenotypes of yeast (Saccharomyces cerevisiae).	*information*, *e-mail*, *industry*, *2017*	*e-mail*, *information*, *learning*, *reading*	*cell*, *addition*, *data*, *analysis*	*Vector (biology)*, *DNA/RNA primer*, *Plant Physiology Journal*, *Primary Structure*

Table 2: Example inputs and corresponding top 4 entity predictions from the models.

entities being recommended. This comes at the cost of lowering prec@k scores, which however correlates with user judgments to a lesser extent, as we showed in Section 3.2. The Random Forest models turn out to be the most competitive. Notice that no linguistic structure is captured in FastXML models, only the bags of entities. This is in contrast with XML-CNN approach which looks at local contexts of feature entities. FastXML performs particularly well on the precision scores, which however is not necessarily useful, as demonstrated in the examples discussed later.

Last, we inspect the sizes of the different models reported in the rightmost column of Table 1. Model size is an important factor to consider in practical applications, e.g. when deploying a system on the device. PFastreXML model takes 150GB, by far the most of all methods, resulting in its capability in recommending tail entities. The *linear models* take 4.7GB related to the fact that in the full $100K \times 100K$ co-occurrence matrix approximately 11% of entries are non-zero. Applying SVD matrix decomposition helps reduce this size significantly. The NN models take around 2GB, significantly less than the random forests.

Analysis To demonstrate the usefulness of the models, in Table 2 we report top 4 entity predictions from XML-CNN and FastXML models for a few example inputs from the test set. Notice how predictions from XML-CNN base are more generic than from XML-CNN IDF. In particular, for the first input related to Antarctic Ice Sheet gaining ice entities *Earth* and *research* are recommended. The relevance of them is clear, however their usefulness is doubtful due to how obvious to the writer they may be. Due to often making such safe predictions XML-CNN base scores higher in precision than XML-CNN IDF. XML-CNN IDF instead makes more specific recommendations, such as *sea level rise* or *temperature* for the first input. This shows how much more beneficial scoring high in NDCG-IDF compared to precision is. An analogous phenomenon takes place between FastXML and PFastreXML – despite FastXML achieving very high precision scores, the predictions tend to be less interesting.

When comparing PFastreXML results against XML-CNN IDF, the results become even more specific. For the first input the entities such as *glaciology* or *Greenland ice sheet* are recommended. For the third input about econometrics and machine learning, both XML-CNN base and FastXML predict a generic entity *information*, XML-CNN IDF predicts more specific *statistics*, and PFastreXML recommends *R programming language*, a popular statistics toolkit for

statisticians and mathematicians with support for machine learning. The usefulness of PFastreXML predictions is particularly profound for the last example about gene biology, where all other approaches back off to very generic entities.

7 Conclusions

In this paper we introduced the problem of entity recommendation for a document writer. Good entity recommendations need to be both relevant and interesting, which we motivated with a user study. In particular, we showed how entities which users write in the document are considered as good recommendations and how IDF score correlates with how useful an entity is considered to be. We corroborated this with example predictions, showing how models scoring higher in metrics weighted by IDF provide more interesting suggestions.

The problem of recommending content to document writers has recently been addressed by industry with tools like Google Explore in Docs and Microsoft Researcher, however the systems are proprietary and the methods have not been published. In particular, this work is the first to formalize the problem and provide insights about it to the research community. We hope that this work will inspire further research on recommending novel content to document writers.

Many avenues for further work can be identified, including: finding a better metric capturing novelty of entities, analyzing the influence of the size of the input passage to quality of predictions, and experimenting with new models. Moreover, recent work has considered incorporating knowledge graph information for better use of entity features (Dalton et al., 2014; Liu et al., 2018). This could also be explored for better feature representation in our problem.

Acknowledgments

We would like to thank Kishore Papineni and Shankar Kumar for inspiring and useful discussions which greatly impacted the shape of this work. We also thank the anonymous reviewers for their insightful suggestions.

References

Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. 1999. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Krisztian Balog, Pavel Serdyukov, and Arjen P. de Vries. 2010. Overview of the TREC 2010 entity track. In *Proceedings of The Nineteenth Text REtrieval Conference, TREC 2010*.

Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. 2015. Sparse local embeddings for extreme multi-label classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 730–738. Curran Associates, Inc.

Zsolt Bitvai and Trevor Cohn. 2015. Non-linear text regression with a deep convolutional neural network. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, pages 180–185.

Ilaria Bordino, Yelena Mejova, and Mounia Lalmas. 2013. Penguins in sweaters, or serendipitous entity search on user-generated content. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, CIKM '13, pages 109–118, New York, NY, USA. ACM.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 335–336, New York, NY, USA. ACM.

Tao Cheng, Xifeng Yan, and Kevin Chen-Chuan Chang. 2007. Supporting entity search: A large-scale prototype search engine. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, SIGMOD '07, pages 1144–1146, New York, NY, USA. ACM.

Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for YouTube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pages 191–198, New York, NY, USA. ACM.

Jeffrey Dalton, Laura Dietz, and James Allan. 2014. Entity query feature expansion using knowledge base links. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '14, pages 365–374, New York, NY, USA. ACM.

Laura Dietz, Manisha Verma, Filip Radlinski, and Nick Craswell. 2017. TREC complex answer retrieval overview. In *Proceedings of The Twenty-Sixth Text REtrieval Conference, TREC 2017*.

Aurélie Herbelot and Eva Maria Vecchi. 2015. Building a shared world: mapping distributional to model-theoretic semantic spaces. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 22–32, Lisbon, Portugal. Association for Computational Linguistics.

Neil Hurley and Mi Zhang. 2011. Novelty and diversity in top-n recommendation – analysis and evaluation. *ACM Trans. Internet Technol.*, 10(4):14:1–14:30.

Himanshu Jain, Yashoteja Prabhu, and Manik Varma. 2016. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 935–944, New York, NY, USA. ACM.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.

Daniel Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 1st edition. Prentice Hall PTR, Upper Saddle River, NJ, USA.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

Carol C. Kuhlthau. 1990. Inside the search process: Information seeking from the users perspective. *Journal of the American Society for Information Science*, 5(42):361–371.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 115–124, New York, NY, USA. ACM.

Zheng-Hao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2018. Entity-duet neural ranking: Understanding the role of knowledge graph semantics in neural information retrieval. *CoRR*, abs/1805.07591.

Nicolas Marie. 2014. *Linked data based exploratory search*. Theses, Université Nice Sophia Antipolis.

K. O. McGraw and S. P. Wong. 1996. Forming inferences about some intraclass correlation coefficients. *Psychological Methods*, 1(1):30–46.

David Milne and Ian H. Witten. 2008. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *Proceedings of AAAI 2008*.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, pages 807–814. Omnipress.

L. Page, S. Brin, R. Motwani, and T. Winograd. 1998. The PageRank citation ranking: Bringing order to the web. In *Proceedings of the 7th International World Wide Web Conference*, pages 161–172, Brisbane, Australia.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *CoRR*, abs/1701.06548.

Yashoteja Prabhu and Manik Varma. 2014. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 263–272, New York, NY, USA. ACM.

David Smith and Jennifer Campbell. 1999. The impact of students' approaches to essay writing on the quality of their essays. *Assessment & Evaluation in Higher Education*, 24(3):327.

Yang Song, Lu Zhang, and C. Lee Giles. 2011. Automatic tag recommendation algorithms for social recommender systems. *ACM Trans. Web*, 5(1):4:1–4:31.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Saúl Vargas and Pablo Castells. 2011. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 109–116, New York, NY, USA. ACM.

Ryen W. White and Resa A. Roth. 2009. *Exploratory Search: Beyond the Query-Response Paradigm*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers.

Hao Wu, Xiaohui Cui, Jun He, Bo Li, and Yijian Pei. 2014. On improving aggregate recommendation diversity and novelty in folksonomy-based social systems. *Personal Ubiquitous Comput.*, 18(8):1855–1869.

Tao Zhou, Zoltán Kuscsik, Jian-Guo Liu, Matúš Medo, Joseph Rushton Wakeling, and Yi-Cheng Zhang. 2010. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Sciences*, 107(10):4511–4515.