Retrieving Skills from Job Descriptions: A Language Model Based Extreme Multi-label Classification Framework

Akshay Bhola[1,3] **Kishaloy Halder**[4*] **Animesh Prasad**[2,3] **Min-Yen Kan**[2,3]
[1] Indian Institute of Technology Kanpur
[2] School of Computing, National University of Singapore
[3] Institute for Applied Learning Sciences and Educational Technology,
National University of Singapore
[4] Zalando SE

`akbhola.bhola@gmail.com` `kishaloy.halder@zalando.de`
`animesh.prasad@u.nus.edu` `kanmy@comp.nus.edu.sg`

Abstract

We introduce a deep learning model to learn the set of enumerated job skills associated with a job description. In our analysis of a large-scale government job portal `mycareersfuture.sg`, we observe that as much as 65% of job descriptions miss describing a significant number of relevant skills. Our model addresses this task from the perspective of an extreme multi-label classification (XMLC) problem, where descriptions are the evidence for the binary relevance of thousands of individual skills. Building upon the current state-of-the-art language modeling approaches such as BERT, we show our XMLC method improves on an existing baseline solution by over 9% and 7% absolute improvements in terms of recall and normalized discounted cumulative gain.

We further show that our approach effectively addresses the *missing skills* problem, and helps in recovering relevant skills that were missed out in the job postings by taking into account the structured semantic representation of skills and their co-occurrences through a Correlation Aware Bootstrapping process. To facilitate future research and replication of our work, we have made the dataset and the implementation of our model publicly available.

1 Introduction

Finding prospective employees with the correct set of skills required for a job is an important aspect of the recruitment process (Brenčič and Pahor, 2019; Lochner et al., 2020). In the current digital age, many recruiters seek to find suitable candidates through multiple channels — *e.g.,* online job portals, professional networks — as well as traditional avenues, such as word of mouth and mass media (Shenoy and Aithal, 2018). In this work, we focus on online job portals (*e.g.,* LinkedIn, Glassdoor), which have emerged as critical players in the job market with millions of active users.

Despite the reach of job portals, finding candidates with the right skill fit still remains challenging. These portals often receive thousands of applications, among which fewer than 10% have the appropriate skills (cielotalent.com, 2014). This bottleneck has attracted both academic and industry researchers from social science and machine learning communities to build automated means to better organize the information present in job descriptions (Boselli et al., 2018). An approach facilitating the matching of jobs to candidates has been to associate a set of enumerated skills from the job descriptions (JDs). Candidate job-seekers can also list such skills as part of their online profile explicitly, or implicitly via automated extraction from résumés and *curriculum vitae* (CVs). Such categorical skills can then be used to induce a match score between JDs and candidate profiles.

We examine the challenge of inferring appropriate job skills from JDs (*cf.*, Fig. 1). JDs consist of a textual description of the job requirements and a list of required skills. The latter helps in indexing JDs to facilitate *ease of search* through facet navigation. We note that there exist crucial skills that *are* mentioned in job requirements (in the job description), but are not listed as required skills (skill labels).

[*] work done while author was at NUS

Job Requirements: Looking for a dynamic individual keen to join a growing organization in the fast-paced and expanding Supply Chain Software Industry. As Singapore goes through the digital revolution, the individual will play a paramount role in developing and upgrading our SaaS tools and valued added services for our clients.

Responsibilities:

1. Work with Architect, Framework Designer to develop the next generation of Cloud base Micro Services Suite
2. Design and implement Micro Service modules primary for Supply Chain systems. Design must be flexible and scalable for any future expansion or upgrade.
3. Establish API services in the API gateway for both internal and external communications and integrations.
4. Document all design works in proper standard formats with detail descriptions. Ensure all design documentations, include module, data are always follow standards and up to date.
5. Conduct Agile development methodology in the Micro Services development life cycle.
6. Carry out unit and integration testing regularly with the QC team.
7. Intensive knowledge on several Java platform technologies, such as JavaEE, DOM/SAX, Annotation, AOP, DI, REST, workflow, etc. Familiar with infra layer technology such as Docker

   .....

Required Skills: .NET, Agile Methodologies, AJAX, ASP.NET, C#, C++, HTML, Java, JavaScript, jQuery, Linux, Microsoft SQL Server, MySQL, Scrum, Software Development, SQL, Subversion, Web Applications, Web Services, XML

Figure 1: Sample job description from the `mycareersfuture.sg` website. Note that *Job Requirements* constitutes the textual description and *Required Skills* constitutes the skill labels of the dataset.

We believe such inconsistencies may be due to the communication gap between the target employer (who are domain experts) and recruiters (who may have little in-domain expertise). To measure the extent of the mismatch, we analyzed JDs from a Singaporean government job portal, `mycareersfuture.sg`. We observe that 40% of JDs miss listing 20% or more explicitly-stated skills in the prose description. In total, 78.86% of such skills were missing.

We propose an automated system to predict the set of required skills given a textual JD. Formally, given an input text $t \in T$, we find a mapping $f : T \rightarrow [0,1]^S$, where $S = |\mathbb{S}|$ and $\mathbb{S}$ is the global skill set. f yields a probability score for each $s \in \mathbb{S}$ labels given t,

$$f(t) = P(r_i = 1|t)$$

where $i \in \{1, \cdots, S\}$, and r_i is the label corresponding to i^{th} skill.

In a nutshell, the task involves labeling the textual descriptions with its corresponding labels selected from a vast set of labels. We approach this problem from a natural language modeling perspective and treat this as an Extreme Multi-label Classification (XMLC) (Lui et al., 2017) task. Our contributions are as follows:

1. We release the large `mycareersfuture` dataset, and provide benchmark results with state-of-the-art baselines to foster research on this important problem[1];
2. We propose a deep learning-based model and a novel Correlation Aware Bootstrapping approach that exploits the correlation between skills and their linguistic footprints;
3. We provide the performance comparison of the proposed methodology with suitable state-of-the-art baseline models.

2 Related Work

We describe the key relevant work from extreme text classification, language modeling and sequence labeling pertinent in developing our approach.

[1] Available at `https://github.com/WING-NUS/JD2Skills-BERT-XMLC`

EXtreme Multi-label Text Classification (XMLC) refers to the text classification scenarios where cardinality of the set of labels is large, i.e., thousands or millions (Liu et al., 2017).

One-vs-All (OVA) (Lui et al., 2017) corresponds to a popular class of approaches for text classification task with high prediction accuracy. This approach is computationally efficient for the XMLC task for modest-sized label sets (up to an order of a few thousand labels).

Embedding-based approaches have also proven effective for this task. They effectively reduce the number of labels by assuming that the label space is well-approximated by a computed low-rank matrix. They reliably learn embeddings for a lower-dimensional label space, then use suitable decompression techniques to map them back to the original label space (Bhatia et al., 2015; Cisse et al., 2013). More recently, methods have been proposed to reduce the information loss during the decompression phase, such as LPSR (Weston et al., 2013) and MLRF (Agrawal et al., 2013).

Following the success of Computer Vision community, deep learning-based approaches have led to a surge of performance in multiple natural language modeling tasks. Neural approaches to natural language processing (e.g., (Kim, 2014)) have also been applied to XMLC tasks (Halder et al., 2018). Methods such as XML-CNN have been proposed, which uses a bottleneck layer to reduce the number of learnable parameters (Liu et al., 2017). A cluster sensitive attention mechanism has also been explored to capture the correlation between labels (Halder et al., 2018), in conjunction with RNN-based text encoders, such as the GRU (Chung et al., 2014). These methods have been used in downstream applications such as tag prediction on Wikipedia articles and news articles, as well as recommending textual articles to potentially interested users.

Language Modeling: Recent developments in language modeling are also relevant to our task. The objective here is to model the syntactic and semantic structure of input language utterances through model training to predict tokens, based on the available contextual information. Models such as ELMo (Peters et al., 2018), Transformer and BERT (Vaswani et al., 2017; Devlin et al., 2019) have significantly improved a bevy of NLP tasks through their unsupervised pre-trained representations.

Another relevant and widely-used approach is **Sequence Labelling**. These tasks involve prediction of a label for each of the tokens present in the textual content. Contiguous chunks of text are then treated as the prediction target. In the Named Entity Recognition (NER) task, entities such as *organization, person, location* are extracted from the text. Typically these methods employ Bidirectional Long Short Term Memory (BiLSTM) stacked with Conditional Random Field (CRF) layers to decode the sequence (Huang et al., 2015; Ma and Hovy, 2016). Context-sensitive character embedding methods also enhance the effectiveness (Akbik et al., 2018).

Although our task can be formulated as a sequence labeling task, we take the multi-label classification perspective for two reasons. First, the label space is finite: most job portal interfaces allow the user to choose a subset from a predefined list of skills in the system. Second, to the best of our knowledge, there is no existing dataset with manually-annotated labels from textual JDs, making it sub-optimal to train a sequence labeling model.

The existing approaches do not address all the unique challenges associated with our skill extraction problem. Alternate techniques of exploiting large scale pre-trained language models for general information extraction tasks have been explored. X-Transformer is a solid representative of such methods (Chang et al., 2020). This BERT-based model first encodes the input query representation, then matches the representation with the clustered labels in an embedding space, formed either by ELMo (Peters et al., 2018) representations or by tf-idf scores. However, obtaining reliable representation for sparse textual label sets featuring domain-specific terminology (*e.g.,* JDs) is challenging. This motivates us to explore alternate approaches – techniques which potentially do not require explicit label representation or predefined semantic meaning of labels. To address these shortcomings, we develop a novel bootstrapping technique to achieve semantic label space mapping without explicitly labelled examples. We aim to define a model that is end-to-end trainable and does not depend on large, domain-specific corpora to learn an embedding space from scratch, in contrast to the higher prerequisite requirements of the X-Transformer.

3 Method

We introduce a neural XMLC framework, **BERT–XMLC**. The architecture is inspired by the models proposed in (Chang et al., 2020) and (Halder et al., 2018). The model comprises two major components:

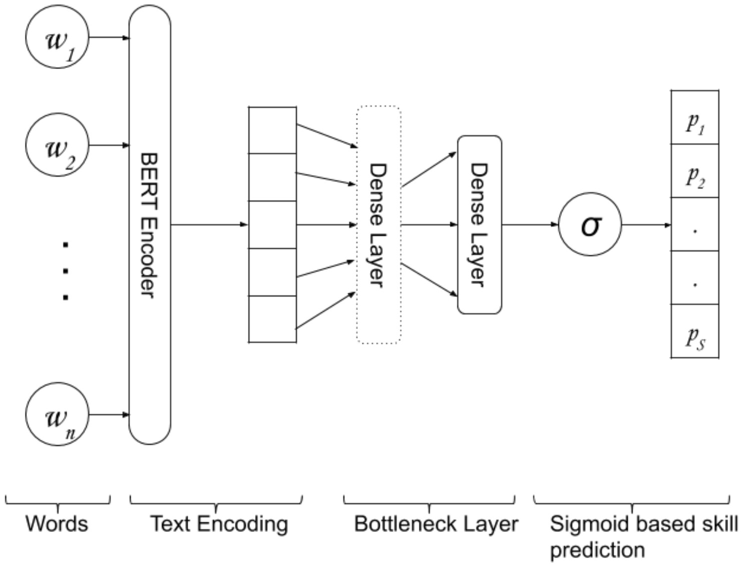

Figure 2: Architecture of our proposed model BERT–XMLC.

1. **Text Encoder:** The model takes the textual JD (t) as input, consisting of n words ($t = \{w_1, w_2, ..., w_n\}$) . We use the pre-trained BERT$_{\text{BASE}}$ model to encode the text into low-dimensional dense vectors (Devlin et al., 2019). Internally, it first looks up the WordPiece embeddings (Wu et al., 2016), then applies a pre-trained bi-directional transformer-based language model to yield a hidden representation of the input. We consider the encoding of the `[CLS]` token as the representation of the textual input.

$$h_t = \text{BERT}_{\text{BASE}}(t) \tag{1}$$

2. **Bottleneck Layer:** Inspired by (Liu et al., 2017), we feed the encoded textual representation to a bottleneck layer. This layer alleviates overfitting by (significantly) limiting the number of trainable parameters. It compresses the encoded representation h_t using, $z_t = tanh(W \cdot h_t + b)$, where W and b are the weight, and bias matrices respectively. Finally, z_t is fed to a fully connected layer using sigmoid activations with S output units, to induce the probability of each possible target label (*i.e.,* skill). Following standard multi-labelling classification approaches, the final layer treats it as S independent binary classification problems. It uses binary cross-entropy loss and optimizes the network using Adam (Kingma and Ba, 2014).

3.1 Correlation Aware Bootstrapping

Let us take a step back and take a closer look at the data. As mentioned, along with the textual JD, we also observe a (rather incomplete) list of required skills. Through our BERT–XMLC model, we are trying to find a relationship between the text and the associated labels (skills). We argue that there are two critical signals which are not explicitly captured in this traditional training method, *i.e,* (a) semantic representation of the skill labels, and (b) co-occurrences of skills. We introduce a bootstrapping process that uses these two assumptions to create artificial training data that guides the initial stages of training.

Semantic Representation of Skills: As shown in Figure 1, the skill labels are essentially phrases in natural language *e.g.,* “Agile Methodologies”, “Software Development” and so on. These phrases (or its constituent words) might also occur in the textual description. Hence, to get a richer representation of the skills, we bootstrap the network with:

$$D_{sem} = \{< s, encode_{OH}(s) > \ \forall\, s \in \mathbb{S}\} \tag{2}$$

where $encode_{OH}(s)$ represents a one-hot encoding for skill s, where $\mathbb{S}$ is the set of all skills. This process yields S additional training points, irrespective of the training data.

Co-occurrence based Correlation of Skills: The network should also learn the correlation between the skills themselves. We direct the network to focus on the skill co-occurrences by additionally training on:

$$D_{corr} = \{< concatenate(s \in \mathbb{S}_k), label(k) > \forall\, k \in \{1, \cdots, M\}\} \tag{3}$$

where $label(\cdot)$ is the original label of k^{th} sample (binary vector), $\mathbb{S}_k$ is the set of all the skills mentioned in the k^{th} sample, and M is the number of training samples. We obtain M additional training points by this method.

We conduct standard training of the model after bootstrapping with these additional $\{D_{sem} \cup D_{corr}\}$ samples, using the original learning objectives. Our bootstrapping process, which we term Correlation Aware Bootstrapping (CAB), sizeably increases the number of training examples from M, to $M + |D_{sem}| + |D_{corr}| = 2 * M + S$. We hypothesize that these added examples help the network to better capture the correlation among skills, which we validate later in our experiments. Note that this bootstrapping method is generic; it can be applied to any off-the-shelf XMLC model for text, as long as the labels have titles that originate from the same vocabulary distribution of input text.

4 Experiments

We have collected data from the Singaporean government website, `mycareersfuture.sg` with permission consisting of over $20,000$ richly-structured job posts having 16 informative fields about the details and the current status of the advertisements. The statistics of the dataset are shown in Table 1. To the best of our knowledge, there is no relevant large-scale publicly available dataset comprising job descriptions with their corresponding list of required skills at the time of conducting the experiments.

For our task, we consider concatenation of “roles & responsibilities” and “job requirements” fields as the textual description, and “required skills” as the set of target discrete labels. We perform the preprocessing operations on the text, inclusive of lower-casing, stopword removal, and rare word removal. We split the dataset into assigned training, validation and testing dataset with an 80:10:10 proportion. To aid the reproduction of our results and to encourage further research in this domain in general, we have made both the code and the dataset publicly available. We might provide updated datasets in the future as more JDs are continuously being posted.

Table 1: MyCareersFuture.sg Dataset (Version 1.0) Statistics.

# of job posts	$20,298$
# of distinct skills	$2,548$
# of skills with 20 or more mentions	$1,209$
Average skill tags per job post	19.98
Average token count per job post	162.27
Maximum token count in a job post	$1,127$

4.1 Baseline Models

To show the effectiveness of different aspects of our approach, we evaluate our model performance against competitive XMLC baselines. These constitute (1) CNN-Kim (Kim, 2014), (2) LSTM (Rocktäschel et al., 2015), (3) BiLSTM (Sun et al., 2017), (4) BiGRU (Halder et al., 2018), (5) BiGRU w/ Cluster Sensitive Attention (CSA) (Halder et al., 2018). For all the RNN-based methods (*i.e.,* Models 2–5, and our proposed model), we fix the architecture to 2 layers to ensure a fair comparison. We implemented our model in PyTorch[2]. Further details about the hyperparameters can be found in our sources.

Metrics: The central theme of this work centers around the concept that the required skills in the ground truth are incomplete. As a result, we do not treat the negatives in the ground truth as true negatives, since they could have just missed out during the manual construction of the JD by the job poster. Hence, we rely on recall-oriented metrics to evaluate our model. We use the following set of metrics:

- **Mean Reciprocal Rank (MRR)** indicates the (reciprocal) position of first true positive in the predicted list of skills. It yields a score between $0 - 1$.
- **Recall@M** captures the recall based on the top-M number of skills in the prediction. This ranges between $0 - 100$.
- **Normalized Discounted Cumulative Gain (nDCG@M)** discounts the true positives that occur later in the prediction rankings. This score ranges between $0 - 100$, similar to recall.

Note that M varies between 5–100 for the metrics of recall and nDCG.

Table 2: Mean Reciprocal Rank (MRR) Comparison.

Model	**MRR**
1. CNN-Kim (Kim, 2014)	0.8195
2. LSTM (Rocktäschel et al., 2015)	0.8417
3. BiLSTM (Sun et al., 2017)	0.8565
4. BiGRU (Halder et al., 2018)	0.8716
5. BiGRU w/ CSA (Halder et al., 2018)	0.8840
6. BERT–XMLC	0.9019
7. BiGRU w/ CSA + CAB	0.8995*
8. BERT–XMLC + CAB (our proposal)	**0.9049**

Table 3: Recall Comparison for various M.

Model	**Recall**				
	@5	@10	@30	@50	@100
1. CNN-Kim (Kim, 2014)	16.38	29.59	59.60	70.40	82.47
2. LSTM (Rocktäschel et al., 2015)	16.83	29.35	57.09	68.65	81.46
3. BiLSTM (Sun et al., 2017)	17.51	31.19	61.77	73.25	84.89
4. BiGRU (Halder et al., 2018)	17.73	31.27	60.84	72.80	84.88
5. BiGRU w/ CSA (Halder et al., 2018)	18.34	32.52	64.19	75.75	87.02
6. BERT–XMLC	19.60	35.58	70.10	80.91	90.26
7. BiGRU w/ CSA + CAB	18.93*	33.84*	66.36*	77.66*	88.28*
8. BERT–XMLC + CAB	**21.67***	**40.49***	**79.59***	**86.60***	**92.24***

* *using proposed methodology*

4.2 Quantitative Analysis

Tables 2, 3 and 4 present MRR, Recall@M, and nDCG@M performance on the skill prediction task. We discuss our observations in two parts.

Effectiveness of BERT–XMLC: Rows 1–6 correspond to the models when they are trained without the bootstrapping steps (cf Section 3.1). We observe that our BERT–XMLC model consistently outperforms

[2] https://pytorch.org/

Boldface results represent best results in corresponding categories

Table 4: Normalized Discounted Cumulative Gain (nDGC) Comparison at various M.

Model	**nDCG**				
	@5	@10	@30	@50	@100
1. CNN-Kim (Kim, 2014)	28.21	40.23	60.60	66.37	71.96
2. LSTM (Rocktäschel et al., 2015)	29.27	40.68	59.43	65.61	71.53
3. BiLSTM (Sun et al., 2017)	30.32	42.77	63.50	69.64	75.04
4. BiGRU (Halder et al., 2018)	30.80	43.11	63.09	69.49	75.09
5. BiGRU w/ CSA (Halder et al., 2018)	31.71	44.62	66.04	72.23	77.46
6. BERT–XMLC	33.64	48.18	71.66	77.45	81.79
7. BiGRU w/ CSA + CAB	32.72*	46.28*	68.33*	74.38*	79.30*
8. BERT–XMLC + CAB	**35.93***	**52.84***	**79.32***	**82.96***	**85.41***

** using proposed methodology*

the baselines by comfortable margins over all metrics. Particularly, in terms of nDCG, we observe a relative improvement of 6.08% for M as small as 5. Unlike the BiGRU with CSA (Row 5), our BERT–XMLC model (Row 6) lacks the additional Cluster Sensitive Attention layer on top of the text encoder. We believe our use of the Transformer model as a base provisions basic attention mechanisms, which partially compensates for the lack of cluster sensitivity.

Effectiveness of Correlation Aware Bootstrapping: Rows 7–8 represent the performance of models which are trained over additional instances imputed by our correlation aware bootstrapping (CAB) process. We observe a surge in model performance, as BERT–XMLC with CAB achieves the best scores consistently with a relative improvement as high as 10.5%, and 6.8% in terms of Recall@5 and nDCG@5, respectively, over its vanilla counterpart (Row 6).

We also note that CAB yields performance boosts consistently even across methods: we observe a jump in the performance for BiGRU with CSA model as well (3.2%, and 3.18% respectively for Recall@5 and nDCG@5). These improvements give empirical evidence that support our hypothesis about the correlation among related skills and skills' intrinsic natural language representation.

4.3 Qualitative Analysis

CAB directs the network to learn the correlation among skills. It is helpful to also visualize this qualitatively on actual instances, in addition to the macroscopic, quantitative improvement. Figure 3 presents a case study sample job advertisement, along with the corresponding gold standard required skills, and the BERT–XMLC & BERT–XMLC+CAB predictions. In the sample, we validate BERT–XMLC+CAB's:

- **Improved recall.** BERT–XMLC with CAB predictions are visibly more accurate compared to BERT–XMLC. Skills such as "Web Services", "C++", "PHP", "Linux" are skills predicted correctly by the bootstrapped model, whereas the vanilla model does not retrieve them.

- **Relevant missing skill prediction.** BERT–XMLC with CAB predicts "Databases" as a required skill, whereas BERT–XMLC does not. Upon manual inspection, we believe that the skill "Databases" is actually relevant and was probably missed out while creating the JD. BERT–XMLC+CAB retrieves this correctly, likely by associating "Databases", and the tokens present in the text such as "SQL" and "databases".

- **Recall of redundant and similar skills.** The JD text has both "SQL" and "databases" tokens, which the BERT–XMLC with CAB likely uses as a basis for predicting –"SQL", "MySQL", "Microsoft SQL Server", "Databases". We believe that the co-occurrence based correlation of skills (*cf* Section 3.1) captured by our model is crucial in predicting semantically similar skills. This is a critical trait for skills recall, as both prospective job seekers and job posters may not use the same terminology to indicate the same skill semantically. We observe that many skills in the top-100 predictions look relevant in our judgement, although they are missing in the ground truth (both in the JD and gold-standard required skill labels).

Job Requirements: Requirements performing end end software development cycle coding using Java j2ee spring framework Oracle pl SQL Multithreading angular js hibernate rest soap api oracle databases shell scripting degree Information Technology Engineering background minimum 5 9 years experience information technology software development must proven experience performing end end software development cycle strong experience coding using java j2ee spring framework strong knowledge angular js hibernate spring rest soap api experience knowledge css html must 4 6 years full stack development frontend design development backend strong knowledge databases good experience agile methodology test driven development self started keen learner new technologies good communication skills

---

Required Skills: Agile Methodologies, Java, Software Development, .NET, C#, C++, HTML, Javascript, jQuery, Linux, Microsoft SQL Server, MySQL, SQL, Web Services, XML, C, CSS, PHP, Python, Software Engineering

BERT–XMLC Predictions: SQL, Java, XML, JavaScript, Software Development, Web Services, Agile Methodologies, MySQL, Microsoft SQL Server, C#, HTML, jQuery, .NET, Web Applications

BERT–XMLC + CAB Predictions: Java, SQL, JavaScript, Software Development, XML, Agile Methodologies, HTML, MySQL, Web Services, jQuery, C#, C++, Linux, Scrum, PHP, Microsoft SQL Server, Web Applications, .NET, CSS, Databases

Figure 3: Sample job description (omitting stop words), with required skills and those predicted by the BERT–XMLC and BERT–XMLC+CAB models.

5 Discussion: On Implicit versus Explicit Skills

We can think of job descriptions and their required skills as two different forms of the same underlying job. The job's manifestation in both forms ideally should corroborate and support each other, but sometimes maybe incongruous. This results in incomplete ground truth in both forms. Our BERT–XMLC+CAB model is architected to overcome this challenge when the two forms do not reinforce each other as expected.

We now analyze how required skills are demographically represented in the JD explicitly (present as substrings in the JD) and implicitly (absent in the JD, but likely inferrable from context). We assess the level of implicitness in the required job skills. As an extreme, if all skills are explicitly mentioned in the JD, our task is easy: it is trivial to list the required skills, as they are all explicit substrings in the JD – a simple string matching algorithm would suffice. At the other extreme, if all required skills are implicit, our task is challenging: every skill needs to be inferred from the context given by the JD. The complexity

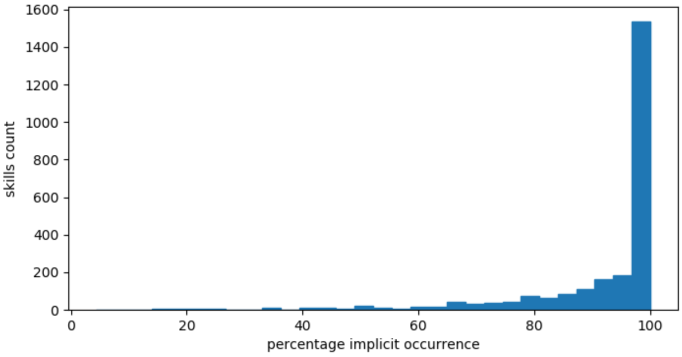

Figure 4: Required Skill Histogram, binned by the percentage of occurrences where it is an implicit skill.

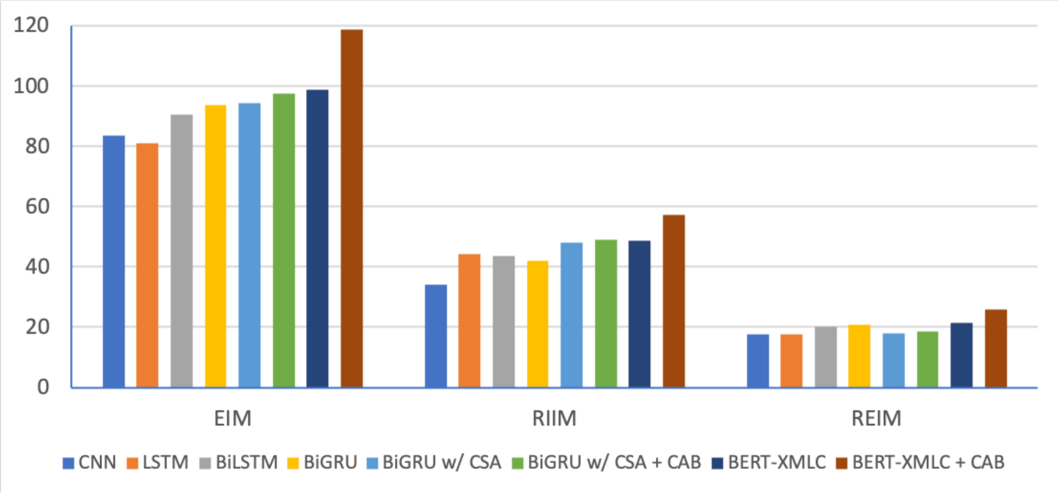

Figure 5: EIM, RIIM, REIM measures over different models.

of the task increases with presence of more implicit skills in the required skill set. Figure 4 shows a macroscopic view of this phenomenon in our dataset through a skill histogram. Each of the 2,548 skills in the dataset is accounted in the figure. We place the skill in one of 30 bins by the percentage of times it occurs as an implicit skill in the JDs where it is required. The average shows that 86.13% of skills are implicit. As the figure is strongly biased towards implicit skills (over 1,500 skills are always implied, never explicitly mentioned), we can see the problem is difficult, and extraction-based models hopelessly fail. Abstractive or generalising approaches such as our model are required for implicit skills.

To study this in more depth, we also propose and plot (Figure 5) the following metrics[3] that capture model performance in a manner sensitive to the implicit and explicit status of skills:

1. **EIM (Explicit Inference Measure):** micro, instance-based measure of explicit skills predicted by the model, compared against gold-standard explicit skills mentioned for a JD. As an example, if the model declares 4 substrings of the JD as skills, and there are 5 explicit skills present in skill labels of JD, EIM = $\frac{4}{5} = 0.8(80\%)$

2. **RIIM (Relative Implicit Inference Measure):** macro, recall-based measure of implicit skills predicted by the model, relative to the entire set of implicit skills. As an example, if the model recovers 2 of 6 skills that are implied but not explicitly substrings in the JD, RIIM = $\frac{2}{6} = 0.33(33.3\%)$

3. **REIM (Relative Explicit Inference Measure):** macro, recall-based measure of explicit skills predicted by the model compared to the entire set of explicit skills.As an example, if the model declares 4 substrings of the JD as skills, and there are total 8 explicit skills present in JD, REIM = $\frac{4}{8} = 0.5(50\%)$

Here, again, the results are consistent: increasing model complexity (leftmost model to rightmost model among the eight models compared) improves both instance-level and macro-level recall. More importantly, these results are also consistent for our BERT–XMLC+CAB model, which performs best in all three metrics. Improvement is most pronounced with the implicit (EIM and RIIM) metrics, clearly showing the improved abstractive generalisation capability of our model.

Note also the EIM measure of BERT–XMLC+CAB tallies to over 100%, suggesting that the model captures a larger number of explicit skills than is actually mentioned in the skill labels (in job descriptions).

[3] Note that these metrics are evaluated based on the skills retrieved by comparing the skill activation with the threshold activation of 0.5.

6 Conclusion

We address the prediction of required skills from job descriptions (JDs). Although job descriptions and their associated required skills can be thought of as two views of the same underlying job, our analysis reveals that many required skills are implicitly signaled. We develop an Extreme Multi-label Classification method that utilizes $\text{BERT}_{\text{BASE}}$ within a Transformer model to predict the required skills from a textual job description. Importantly, we propose a novel bootstrapping approach that exploits the underlying natural language representation of skills and their co-occurrence relationships with other skills. We perform experiments on a large real-world dataset from a popular Singaporean government job portal, `mycareersfuture.sg`. We show that our model outperforms recent competitive baselines, especially in recalling such implicit skills.

References

Rahul Agrawal, Archit Gupta, Yashoteja Prabhu, and Manik Varma. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd international conference on World Wide Web*, pages 13–24. ACM.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA, August. Association for Computational Linguistics.

Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. 2015. Sparse local embeddings for extreme multi-label classification. In *Advances in neural information processing systems*, pages 730–738.

Roberto Boselli, Mirko Cesarini, Stefania Marrara, Fabio Mercorio, Mario Mezzanzanica, Gabriella Pasi, and Marco Viviani. 2018. Wolmis: a labor market intelligence system for classifying web job vacancies. *Journal of Intelligent Information Systems*, 51(3):477–502.

Vera Brenčič and Marko Pahor. 2019. Exporting, demand for skills and skill mismatch: Evidence from employers' hiring practices. *The World Economy*.

Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit Dhillon. 2020. Recognizing text entailment via bidirectional lstm model with inner-attention. In *SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.

cielotalent.com. 2014. So Long, Traditional Job Boards? `https://www.cielotalent.com/insights/talent-acquisition-fast-facts-so-long-traditional-job-boards`. Online; accessed 25 September 2019.

Moustapha M Cisse, Nicolas Usunier, Thierry Artieres, and Patrick Gallinari. 2013. Robust bloom filters for large multilabel classification tasks. In *Advances in Neural Information Processing Systems*, pages 1851–1859.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. ACL, June.

Kishaloy Halder, Lahari Poddar, and Min-Yen Kan. 2018. Cold start thread recommendation as extreme multi-label classification. In *Companion of the The Web Conference 2018 on The Web Conference 2018*, pages 1911–1918. International World Wide Web Conferences Steering Committee.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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

Benjamin Lochner, Christian Merkl, Heiko Stüber, and Nicole Gürtzgen. 2020. A note on recruiting intensity and hiring practices: Cross-sectional and time-series evidence.

Jingzhou Lui, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In *International ACM SIGIR Conference on Research and Development in Infromation Retrieval (SIGIR)*, pages 115–124. ACM.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany, August. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June. Association for Computational Linguistics.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*.

Varun Shenoy and PS Aithal. 2018. Literature review on primary organizational recruitment sources. *International Journal of Management, Technology, and Social Sciences (IJMTS)*, 3(1):37–58.

Chengjie Sun, Yang Liu, Chang'e Jia, Bingquan Liu, and Lei Lin. 2017. Recognizing text entailment via bidirectional lstm model with inner-attention. In *International Conference on Intelligent Computing*, pages 448–457. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Jason Weston, Ameesh Makadia, and Hector Yee. 2013. Label partitioning for sublinear ranking. In *International conference on machine learning*, pages 181–189.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.