LLM Approaches to Job Title Classification: A Midterm Review

Corey Sarcu

Georgia Tech

Project Overview

- Netskope, a global cybersecurity and software company, is interested in the exposure of their public-facing content (both digital and in-person events) to their target customer segments.
- Attendees fill in personal information, which includes their job title.
- Netskope then categorizes each job title into a suitable job function, job role, and job level to determine to what degree each attendee is involved in information security functions, as well as how senior they are in their department.
- Using this information, Netskope aims to discern the penetration of their communications into key industry sectors, as well as identify relevant attendees for follow-up marketing of services and sales initiatives.

Netskope logo [1]

Georgia Tech

Current Solution

- Netskope currently uses a keyword lookup table, where they check to see if certain keywords are contained in the job title, to which they hard-assign a function, role, and level, subject to keyword priority.
  - For example, the keyword(s) “chief tech” are given a priority of 10 and associated with the role of “IT General”. The keyword “security” has a priority of 2, which outranks “chief tech” and is associated with a role of “Information Security”. The presence of “security” will overwrite the influence of “chief tech” in the scenario where both are present, resulting in a role of “Information Security”.
- This can lead to logic sequences that produce unpredictable contradictions and changes in the way job titles were tagged prior.
- Keywords are a clunky solution to the extent that all permutations of suitable words with different prefixes/suffixes and use of spaces will have to be handled individually as separate hard-coded entries.
- The assessment of keyword priority is entirely arbitrary and leads to a system of rules whose internal logic is obscured from the user.

Georgia Tech

The Data

- Information gathered from ICP (Ideal Customer Profile) leads
- Title (text, the attendee's job title)
- Job Function (text, a broad identifier for determining the department the attendee works in)
- Job Role (text, identifies the attendee's specific area of focus within their department)
- Job Level (text, the seniority of the attendee)
- The "Title" field is an input; the other fields are all outputs generated from the keyword methodology at the time of data gathering.

The Task

- Develop a more robust modeling solution that builds on the labeling generated by the prior keyword methodology that is able to generate functions, roles, and levels for a given title input.
  - The model should be able to somewhat accurately reproduce the results of the keyword method, but may (and should) diverge in cases where the keyword method has inconsistencies due to the interaction of various keyword priority rules.
- Function is most important to predict accurately, followed by role, and then by level.
- Take model result and cast into Netskope's new desired hierarchical classification logic – this differs from past logic.
- Labels and outputs will all be in English.
- If time allows, explore and explain insights produced by the model.
  - Examples: new keywords not previously known, a quantification of priority, or other data quirks

Exploratory Data Analysis (EDA)

EDA

- 1.6 million records
  - Over 330k unique titles, 90 unique functions, 42 unique roles, and 56 unique levels
- Function
  - Most common is IT, accounting for 64% of the data.
  - Most common 5 categories account for 90% of the data.
- Role
  - Most common is Information Security, accounting for 28% of the data.
  - Most common 5 categories account for 91% of the data.
- Level
  - Most common is Contributor, accounting for 39% of the data.
  - Most common 5 categories account for 99% of the data.
- Records with any one of the desired outputs that are NA/null account for 7% of the data.
- Records with titles being NA/null account for 3% of the data.
- Records with any NA/null in either title or outputs account for just over 7% of the data.
- There are just under 64k unique words in the title field, with only 112 of these being stopwords such as a, an, the, etc.

Observations

- The function field is the most skewed of the output categories, but not to the degree that any model structure augmentations have to account for it. If need be, resampling can be done on the under-represented categories to compensate for this.
- NA/null values are not very numerous and can be dropped from the data without greatly affecting trained model quality.
- Significant dimension reduction/grouping needs to occur on the output categories to make the classification problem more tractable.

Extract, Transform, and Load (ETL)

ETL

- Function – reduced to 5 categories: IT, Non-ICP, Engineering, Procurement, and Risk/Legal/Compliance
- Role – reduced to 6 categories: Non-ICP, Information Security, Networking, IT General, Systems, and Governance Risk Compliance
- Level – reduced to 6 categories: Contributor, Manager, Executive, Director, C-Level, and Unknown
- Any records with any one of the 4 fields coming through as NA or null are dropped from the final record set for modeling.
- Because I am building a deep learning model in pytorch, the data will need to be set up to load in through a batched dataloader so as to limit the amount of data being processed through the graphics card (significantly speeds up deep learning computations).

Literature Review

Georgia Tech

Literature Review

- Transformer models are useful for a variety of language inference tasks. They consist of an encoder block that allows for processing of inputs of varying lengths, and (optionally) a decoder block that can produce outputs of varying lengths [2].
- For our classification task, because the outputs of each sequence are fixed in length (each sequence leads to a classification vector for function, role, and level), we only need to make use of the encoder block.
- We can take advantage of a pre-trained publicly-available model to fine-tune for our task, even when the pre-training was on a completely different task [3].
- BERT, a popular and widely-available transformer encoder model, has been effective in fine-tuning to a wide-ranging array of language processing tasks. It is bi-directionally encoded, which means that its pretraining tasks include predicting masked words based on words both before and after it in the sequence [4]. Its base model has 110M parameters [5].

Literature Review (continued)

- DistilBERT uses knowledge distillation to achieve 97% of BERT performance with only 60% of the model parameters (66M in the base model [5]), thus running significantly faster training and inference loops [6]. Considering that our task is much simpler than most of BERT's benchmark tasks, DistilBERT should be more than sufficient for our particular sequence classification problem.
- Time provided, if there is enough potential for marginal improvement that might be afforded by larger encoder models, the full BERT model can be fit. If the opposite is true, and computing resources prove to be a serious limitation, ALBERT [7] has only 11M parameters in its base model, and thus may run significantly faster than even DistilBERT [5].

Procedure and Experiments

Georgia Tech

Procedure and Experiments

- Starting with a single pre-trained DistilBERT uncased model, I can simply take the outputs from that model and feed into 3 separate classification layers in parallel – for function, role, and level, respectively. Example code is shown in [8].
- The pre-trained model should be a good initialization for the upstream classification task. I will allow all of the parameters in the model to update – both the layers that I added as well as the pre-trained model.
- I will start with cross entropy as a loss function for gradient backpropagation through the model; if I see significant underperformance due to the skew in output classes (most notable with the function output), I may try resampling [9], but only if strictly necessary.
- If it appears that there is significant room for improvement in the model, I can try hyperparameter tuning to further improve initialization conditions for model training.

Procedure and Experiments (continued)

- Although we are trying to predict 3 different labels, I will build a single loss function based on a weighted average of cross entropy losses for each label. The weights will be based on 1/accuracy, where accuracy represents the training accuracy of the last 1000 batches for the given label. This way, labels that are less accurate will get a larger weight in the loss function, and parameter updates associated with these more inaccurate labels will be larger in magnitude.
- Weights for validation will be calculated similarly, but with accuracy over the entire training dataset, not just the latest 1000 batches.

Training Settings

- Max sequence length – 64 tokens (some sequences are truncated because of this, but job titles longer than 64 tokens are assumed to carry relevant information closer to the beginning)
- Batch size – 128 samples
- Epochs – 10
- Learning rate – 1e-05
- Loss function weights – role/function/level will all start at 1 and update as the model trains
- Stop early if all 3 labels reach at least 90% accuracy during any epoch
- Model checkpoint after each epoch
- Data split 80/10/10 training/validation/test; validation data metrics examined after each epoch to determine best checkpoint

Overview of Preliminary Results

Georgia Tech

Overview of Preliminary Results

The results shown to the right correspond to epoch 8 out of 10. This means that the model is already predicting function labels accurately more than 97% of the time on out-of-sample data, for example. Each epoch took about an hour to run in total. Based on these results, it seems unlikely I'll need to run different model structures, or tweak hyperparameters, as these results are already quite strong. The fact that validation results are not significantly worse than training results rules out concerns of any significant overfitting in this model.

		Training	Validation
	Loss	0.4713	0.519
Accuracy	Function	0.979	0.9764
	Role	0.8868	0.8832
	Level	0.9485	0.9463

Georgia Tech

Ranking of Training Predictions by Loss, Descending

```python
# Now sort by loss descending to look at the observations that the model gets most wrong
inference_results_sorted = inference_results.sort_values(by='Loss',ascending=False)[['Title','Job Role_Text','Job Function_Text','Job Level_Text',
                                                                                     'Job Role Predicted_Text','Job Function Predicted_Text',
                                                                                     'Job Level Predicted_Text','Loss']]
inference_results_sorted.head(20)
```

Python

	Title	Job Role_Text	Job Function_Text	Job Level_Text	Job Role Predicted_Text	Job Function Predicted_Text	Job Level Predicted_Text	Loss
272265	Chief Information Security Officer	IT General	IT	Director	Information Security	IT	C-Level	21.508377
1002518	Cto	Networking	IT	C-Level	IT General	IT	Executive	20.611830
254623	DX???????	Systems	IT	Contributor	Non-ICP	Non-ICP	Manager	15.255075
166917	Security Operations Manager	Networking	Non-ICP	Manager	Information Security	IT	Manager	15.049135
713354	Security Operations Manager	Networking	Non-ICP	Manager	Information Security	IT	Manager	15.049135
476769	Issm	Information Security	Engineering	Contributor	Non-ICP	Non-ICP	Manager	14.638029
251121	Security	Governance Risk Compliance	IT	Manager	Information Security	IT	Contributor	13.821310
574676	Security	Governance Risk Compliance	IT	Manager	Information Security	IT	Contributor	13.821310
955758	Dirección General	Non-ICP	Non-ICP	Director	Information Security	IT	C-Level	13.527388
290703	Dirección	Non-ICP	Non-ICP	Director	Information Security	IT	C-Level	13.114651
1042937	Isa	Information Security	IT	Contributor	Non-ICP	Non-ICP	Contributor	13.025041
869919	Head Of It Service Management	Governance Risk Compliance	IT	Executive	Information Security	IT	Executive	12.945917

Ranking of Training Predictions by Loss, Descending (continued)

- To produce the previous slide's output, I ran model predictions on the training set, using the weights resulting from the training model accuracy of the epoch leading up to the model checkpoint.
- Digging into these results, I discovered that there are labeling inconsistencies in the data – for example, the very top result in the table (where the loss is highest, i.e. where the model prediction is "most wrong") is on a title "Chief Information Security Officer". The predicted level was C-Suite, but the actual level was Director. As it turns out, this title is not labeled consistently in the data – in the vast majority of cases, this is labeled as C-Suite, but this particular observation is labeled as a Director. Logically, the data appears in error on this point, and the model prediction, while "wrong" relative to the data, appears to be right according to common sense. This level of inconsistency makes the modeling task difficult, so more data cleaning is needed before developing a reliable model.
- Also, while these are supposed to be modeling/predicting in English, there are very clearly foreign languages present here (ex.: Dirección). These will need to be removed to stay within the assigned task scope.

Further Data Cleaning

Title	Job Function	Job Role	Job Level	Record_Count	Instance_Count	grouping_size
AE	Sales	None Technical	Executive	230	2	2
ANALYST	IT	None Technical	Contributor	2	2	1
ANALYST	Unknown	None Technical	Contributor	2	2	2
ATAE	Other	None Technical	Contributor	2	3	2
ATAE	Unknown	None Technical	Contributor	13	3	1
ATAE	Unknown	None Technical	Unknown	1	3	3
AVP	Management	IT General	Executive	4	4	4
AVP	Management	Information Security	Contributor	50	4	2
AVP	Management	Information Security	Executive	5	4	3
AVP	Management	None Technical	Executive	56	4	1
AVP - Technology	IT	IT General	Executive	1	2	2
AVP - Technology	IT	Information Security	Contributor	2	2	1

	A	B	C
	Title	var_record_count	total_record_count
1	VP of IT	14620.5	1893
5	Cio	32258	2032
5	IT	40618.44444	1019
7	Cyber Security Engineer	79769.64286	1102
7	Director Of It	99077.5	1230
1	Partner	104779	1041
3	It Director	114856.2143	1694
9	Director Of Information Technology	115021	1302
0	Managing Director	115525	1179
1	Network Security Engineer	119975.1429	1188
4	VP	125416.1429	1469
7	Cyber Security Analyst	133959.25	1134
3	IT Specialist	136963.5833	1281

- I provided Netskope with 3 pieces of information:
  - Examples of foreign language titles
  - List of titles that had multiple function/role/level assignments, and the number of records of each (snippet shown on top left)
  - List of titles and the statistical variance of record counts across different function/role/level combinations (snippet shown on bottom left, filtered on total record count > 1000)

Georgia Tech

Further Data Cleaning (continued)

- To combat the foreign language issue, Netskope re-filtered the data to only capture labels from US-based sources. The new data is about half the size of the original at just over 860k records. The tables on the previous slide were based on this new reduced dataset.
- The table showing the record counts for each title/function/role/level grouping are ranked by record count in the grouping_size field, with 1 associated with the largest group. I suggested to Netskope it might be easiest to just re-label all of the records associated with each title to be the same as the largest group by number of records, assuming that this was indeed the correct label. This will re-label about ~70k records, or 8% of the overall data.
- The statistical variance table indicates that titles with lower variance in their record counts are more uncertain, since this means that the number of records in each different labeling combination are more similar and there is not as much of a prevailing “favorite”.

Next Steps

Georgia Tech

Next Steps

- More data cleaning to optimize chance of model success
- Re-fitting model on optimized data
- Evaluate final model metrics on test set
- GradCAM or some other method to rank importance of input tokens [10]
  - This will inform us on keywords that might not currently be in the provided keyword list
- Production-level model to be used for inference – feed in a .csv file of titles, output a fitted function, role, and level
  - This will need to take in the outputs from my model and slot them into the desired go-forward function/role/level hierarchy provided by Netskope, which differs from the way things were categorized historically.
  - CSV output subject to change based on format desired by Netskope – this has not yet been specified

Gantt Chart

	Feb	March				April		
	23 - 29	1 - 7	8 - 14	15 - 21	22 - 28	29 - 4	5 - 11	12 - 18
Data Cleaning								
Model Refitting								
GradCAM/Keyword Importance								
Productionalizing Model								
Final Report								

- Data cleaning – adding an extra week in case there are unforeseen data issues, and also assuming some more back-and-forth with Netskope
- Keyword importance and setting up model for production – assuming there will be back-and-forth with Netskope, so allowing extra time for this
- Deadlines
  - Final report to Netskope on April 11th
  - Final report to Georgia Tech on April 18th
- I am working as an individual, so I will be completing all this work myself

Georgia Tech

References

References

1. Netskope. (2024, February 16). *Netskope Logos*. Netskope. https://www.netskope.com/company/newsroom/netskope-logos
2. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). Attention Is All You Need. *arXiv [Cs.CL]*. Retrieved from http://arxiv.org/abs/1706.03762
3. Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. https://doi.org/10.18653/v1/p18-1031
4. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR, abs/1810.04805*. Retrieved from http://arxiv.org/abs/1810.04805

Georgia Tech

References (continued)

5. *Pretrained models*. Huggingface. (n.d.). https://huggingface.co/transformers/v4.4.2/pretrained_models.html
6. Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR, abs/1910.01108*. Retrieved from http://arxiv.org/abs/1910.01108
7. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *CoRR, abs/1909.11942*. Retrieved from http://arxiv.org/abs/1909.11942
8. Taunk, D. (n.d.). *Fine Tuning DistilBERT for MultiLabel Text Classification*. Google Colab. https://colab.research.google.com/github/DhavalTaunk08/Transformers_scripts/blob/master/Transformers_multilabel_distilbert.ipynb

References (continued)

9. Burnaev, E., Erofeev, P., & Papanov, A. (2015). Influence of resampling on accuracy of Imbalanced Classification. *SPIE Proceedings*. https://doi.org/10.1117/12.2228523
10. Jeon, H. (2018, July 25). *Let sentiment classification model speak for itself using Grad Cam*. Medium. https://medium.com/apache-mxnet/let-sentiment-classification-model-speak-for-itself-using-grad-cam-88292b8e4186