Automatic Occupation Coding using Deep Learning

Ravikiran Chanumolu
International Institute of Information Technology, Hyderabad
ravikiran.chanumolu@research.iiit.ac.in

Mihir Shekhar
Microsoft India
mihir.shekhar@microsoft.com

Kamalakar Karlapalem
International Institute of Information Technology, Hyderabad
kamal@iiit.ac.in

***Abstract*—Classification of occupations plays an important role in a variety of activities such as matching job seekers with job vacancies, administration of worker's compensation, and management of employment-related migration. Classification of occupations also referred to as Occupation Coding, is the categorization of individual occupations based on the kind of work and skill-level involved. Occupation Coding is an important step in the collection and dissemination of statistics from sources such as population censuses, labour market surveys and employer surveys. Historically, the task of classifying jobs to standard classifications was done manually. However, the drawbacks of manual coding have led researchers to develop automatic methods for occupation coding. In this paper, we compare and contrast different deep learning approaches for classifying jobs into occupational categories. We show these comparative results using a large data set of 720k job descriptions. Further, previous approaches based on rule based and hybrid methods only use the job titles for classification. The success of deep neural networks in learning from high-dimensional data allows for utilizing full job description in addition to job title for finding O*NET code. We show that using entire job description significantly improves classification accuracy.**

***Index Terms*—Occupation Coding, Deep Learning, Hierarchical Attention Network, O*NET**

I. Introduction

Occupation Coding is an important tool to match job seekers with vacancies in the industry. It plays a vital role by assisting job seekers to find jobs that align with their skill-set and provide appropriate remuneration. Moreover, the tool is widely essential in official statistics and social science research. For instance, classifying occupation aids in comparing the wage gap among different genders across occupations. It also helps relate the social and economic status of the individual to the occupation. In addition, since the occupation is a risk factor in many diseases and injuries, the tool facilitates industrial hygiene and epidemiological analyses.

There are many Occupation coding systems such as Standard Occupation Coding (SOC), German National Occupation System, Holland Occupational Themes, etc. Our work focuses on the Occupational Information Network (O*NET) which is the most widely used Occupation classification system in the United States [1]. O*NET contains descriptions of hundreds of occupation categories detailing the job characteristics and requirements. Each occupation category in the O*NET system is assigned a code which has a hierarchical format. For instance, the code 11-2011.01 represents the O*NET occupation 'Green Marketers', which is a subset of broad occupation 'Advertising and Promotions Managers' (11-2010), which in turn belongs to the minor group 'Advertising, Marketing, Promotions, Public Relations, and Sales Managers' (11-2000), that is encompassed by major group 'Management Occupations' (11-0000).

Commonly used coding techniques use rule-based approaches [1] which require constant efforts, expert knowledge and manual analysis for updating the rule set because job descriptions and O*NET taxonomy change with time. This issue can be solved with supervised machine learning approaches [2], [3] that use Naïve Bayes, Support Vector Machines (SVM) or K-Nearest Neighbors (KNN). However, high dimensionality of job descriptions and a large number of O*NET classes result in poor accuracy when traditional machine learning methods are used. Hybrid models [4], [5] that combine rule-based and machine learning approaches give better results as they incorporate expert knowledge. Nevertheless, they suffer from the same disadvantages as rule-based approaches. Additionally, these models were developed for other occupation systems, hence the rules are not compatible with the O*NET system. On the other hand, deep learning techniques have proven to give a state-of-the-art performance in a variety of text classification problems [6]–[8]. They are capable of learning good feature representations of high dimensional textual documents that improve classification accuracy. In our work, we use Hierarchical Attention Network (HAN) proposed in [9] to classify job descriptions into O*NET categories at various levels in the hierarchy.

As detailed above, O*NET classes have a hierarchical format. The hierarchical relationship between classes has been exploited previously to improve the accuracy of prediction [10], [11]. Recently, Xiong et al. [12] used a variant of HAN to classify Shopify app descriptions to hierarchically-structured app categories. It incorporated the taxonomy prior knowledge, via the use of hierarchical output layers representing different levels of the taxonomy. The proposed HAN variant outperformed the HAN model with a single output layer. Since our work is focused on a similar dataset, we used the HAN variant to include O*NET taxonomy in model training. However, we observed reduced performance when using the HAN variant as compared to the model that ignored the class hierarchy, therefore contradicting the findings of Xiong et al. [12]. We attempt to qualitatively explore some characteristics of O*NET taxonomy that explains this contradiction.

The major contributions of this paper are:

- We compare and contrast different deep learning models on classifying job descriptions into O*NET categories.
- Unlike previous approaches we use the complete job description for Occupation Coding, thereby facilitating bet-

[1] http://www.onetonline.org/

ter fine-grained classification on similar O*NET classes. It was observed that utilizing the complete job descriptions improved the classification accuracy by 7%.
- We perform extensive experiments on a large dataset of 720K job descriptions and show the efficacy of various deep learning architectures.
- We explore the probable reason for the decrease in classification accuracy when O*NET hierarchy is incorporated in model training.

II. Related Work

Occupation Coding techniques can be broadly divided as manual, computer assisted and automated. Manual methods, while accurate, are time-consuming and expensive [13], [14]. Computer assisted techniques such as Cascot [15] use software to filter codes, from which a human selects the best option. While such computer assisted approach alleviates some of the problems with manual coding, they still require human in the loop.

Most automated coding techniques use rule-based approaches. Gweon et al. [2] and Schierholz et al. [5] use a database of job titles and their codes. If the new job title exists in the database, then the corresponding code is assigned. Similarly, the U.S. Information Technology Support Center (ITSC) produced the OccuCoder software [2] for coding occupations to the O*NET system. OccuCoder uses word matching with the O*NET database. Instead of exact match, other approaches [1], [3] use language similarity to find partial matches. Dictionary-based Occupation Coding is another rule-based technique commonly used by organizations. The dictionary contains occupation codes and keywords which include unigrams, bigrams and trigrams in job titles. Each entry in the dictionary is weighted based on the relative frequency. For instance, Automatic Coding by Text Recognition (ACTR) [16] is a dictionary-based software created by Statistics Canada to code survey responses to standard classifications. Rule-based approaches are easy to interpret and verify. However, taking into account the complexity of Occupation Coding, rule-based approaches require a lot of time, analysis and expert knowledge to generate the rules.

Previous works have also explored supervised machine learning approaches. Kirby et al. [3] used the Naïve Bayes algorithm for coding Scottish occupation records to Historical International Standard Classification of Occupations (HISCO). Schierholz et al. [5] used Naïve Bayes and boosting methods to code job titles in the ALWA Survey to German national occupation classification. Javed et al. [4] created an occupation title classification software called Carotene. The system has a hierarchical structure. Support Vector Machine (SVM) is used for coarse classification in the first level of the SOC-O*NET classification system and K Nearest Neighbours (KNN) is used for fine-grained classification for the lower levels of SOC-O*NET.

[2]http://www.itsc.org/Pages/pub_aocode.aspx

Interestingly, deep learning architectures have not been explored for this important problem of occupation coding. To the best of our knowledge, we present the first work on comparing various deep learning architectures for this task.

III. O*NET coding using Job Description

In the section, we elaborate on using HAN and HAN with hierarchical outputs for occupation coding. We also provide a brief overview on Long Short Term Memory (LSTM), Convolutional Neural network (CNN), and Recurrent Convolution Neural Network (RCNN) that serve as baseline approaches.

A. Hierarchical Attention Network (HAN)

We classify job descriptions into O*NET categories, using HAN [9], which hierarchically models the structure of a document. Let a document D be composed of N sentences $(s_1, s_2, ..., s_N)$, and each sentence comprises of sequence of words $[w_{i_1}, w_{i_2}, , ..., w_{i_{T_i}}]$ where T_i is the number of words in sentence s_i. We first embed the words to vectors using an embedding matrix, W_e. Let the word vector corresponding to the word w_{i_t} be x_{i_t}. We then obtain the representation for the given word w_{i_t} using the GRU network as follows.

$$\overrightarrow{h_{i_t}} = \overrightarrow{GRU}(x_{i_t})$$
$$\overleftarrow{h_{i_t}} = \overleftarrow{GRU}(x_{i_t})$$
$$h_{i_t} = \overrightarrow{h_{i_t}} \oplus \overleftarrow{h_{i_t}}$$

where $\oplus$ represents vector concatenation and $\overrightarrow{h_{i_t}}$ and $\overleftarrow{h_{i_t}}$ represent forward and backward hidden states at the t^{th} word. Since different words in a sentence have different contribution to its overall meaning, the vector representation of a sentence is computed as a weighted sum of the word representations. Attention mechanism is used to extract these weights. The word-level representations are passed through a one-layer MLP to get u_{i_t} as the hidden representation of h_{i_t}. The importance of the word is then measured as the similarity of obtained u_{i_t} with a word-level context vector u_w to get a normalized importance weight α_{i_t} through a softmax function as given below.

$$u_{i_t} = tanh(W_w h_{i_t} + b_w)$$
$$\alpha_{i_t} = \frac{\exp(u_{i_t} u_w)}{\sum_t \exp(u_{i_t} u_w)}$$
$$s_i = \sum_t \alpha_{i_t} h_{i_t}$$

The word context vector u_w is randomly initialized and jointly learned during the training process.

Using the sentence vectors s_i $(i \epsilon [1, L])$, the document representation is computed using a similar approach. The sentences are first encoded using the GRU network.

$$\overrightarrow{h_i} = \overrightarrow{GRU}(s_i)$$
$$\overleftarrow{h_i} = \overleftarrow{GRU}(s_i)$$

$$h_i = \overrightarrow{h_i} \oplus \overleftarrow{h_i}$$

Finally, the sentence level context vector u_s along with attention mechanism is used to find the final document representation v as given below.

$$u_i = tanh(W_s h_i + b_s)$$

$$\alpha_i = \frac{\exp(u_i u_s)}{\sum_t \exp(u_i u_s)}$$

$$v = \sum_i \alpha_i h_i$$

Document vector v is then used for document classification:

$$p = softmax(W_c v + b_c)$$

Negative log likelihood of the correct labels is used as training loss:

$$L = -\sum_d \log p_{dj}$$

where j is the ground truth label of document d.

B. Hierarchical Attention Network with Hierarchical Outputs

To utilize the hierarchical structure of O*NET categories in model training, we also experiment with a variant of HAN proposed in [12]. The HAN variant is different from the HAN model in the following two ways.

- It includes hierarchical output layers depicting label taxonomy.
- It eliminates the one-layer MLP in the attention-building part of the networks. The attention weights are directly calculated as the similarity of context vector and the output hidden state vectors of the GRU.

In the new architecture different levels of taxonomy labels are represented by multiple output layers. The first output layer fully connected to the document representation corresponds to the bottom (or leaf) taxonomy level (level M). The first output layer is as follows.

$$p_M = sigmoid(w_c v + b_c)$$

where w_c and b_c represent weights and biases in a fully connected layer respectively. The next output layer is derived from the current output layer by using average pooling.

$$p^k_{m-1} = avg_j p^j_m$$

where the taxonomy node j at level m is a child of taxonomy node k at level $m-1$. The loss function at level m is a sigmoid cross entropy loss, which is as follows.

$$J_m = -\sum_{d=1}^{N}\sum_{k=1}^{C_m}(y^k_m \log p^k_m + (1 - y^k_m)\log(1 - p^k_m))$$

where C_m is the size of taxonomy at level m, which is also the dimension of the output layer that represents level m of the taxonomy. Here N is the total number of training instances. For a document d, y^k_m is 1 if the document belongs to category k at the level m in the hierarchy else it is 0. Note that we have ignored the d subscript from y and p variables to avoid clutter. The overall training loss:

$$J = \sum_{m=1}^{M} w_m J_m$$

where w_m is the weight on level m. The weights w_m are hyper parameters of this model.

Long Short Term Memory (LSTM) Networks

LSTM [6] captures long term dependency among word sequences over a longer time period. The LSTM architecture is primarily composed of a memory cell, an input gate, an output gate and a forget gate. It is a chain constructed by repeating modules of neural networks. While the memory cell stores information that runs across the whole chain, the three gates are designed to control whether to add or block the information to the memory cell. The gates regulate the update of the current memory cell and hidden layer H_t with the output from the previous hidden layer H_{t-1} and the current input x_t. The input gate manages the addition of information in the memory cell while the forget gate represents the portion of the information that will be kept. This is summarized in the following equations.

$$i_t = \sigma(w_i[H_{t-1}, x_t] + b_i)$$

$$f_t = \sigma(w_f[H_{t-1}, x_t] + b_f)$$

$$o_t = \sigma(w_o[H_{t-1}, x_t] + b_o)$$

$$\tilde{c_t} = \tanh(W[H_{t-1}, x_t] + b)$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c_t}$$

$$H_t = o_t \cdot \tanh(c_t)$$

$\sigma(.)$ represents sigmoid function. At the current time t, H_t indicates hidden state; i_t refers to the input gate, f_t represents the forget gate, and o_t denotes the output gate. w_i, w_f, and w_o represent the weights of these three gates, while b_i, b_o, b_f refer to the biases of the gates respectively.

Convolutional Neural network (CNN)

CNNs are deep learning models usually used to build models for images. But they have also been used for text processing as well. The input to a CNN [8] is a text sequence containing n words $[w_1, w_2, ..., w_n]$. We embed individual words into a d dimensional vector. This results in $d \times n$ matrix which is passed through a convolution layer with m filters. A sliding window of length l is passed over the sequence which converts the dxn matrix to nxm matrix. Then, max pooling is applied to get m dimensional vector p that is independent of sequence length. Finally, this fixed-length vector is processed using the Rectified Linear Units (ReLU) non-linearity.

Recurrent Convolution Neural Network (RCNN)

RCNNs are deep learning models which combine the characteristics of Recurrent Neural Networks (RNNs) and CNNs. The input to an RCNN [7] is a short text S, which contains a sequence of words $[w_1, w_2, ..., w_n]$. The embedding vector of the text sequences is passed through a bidirectional LSTM. Then for each word w_i in a text sequence, the final embedding vector x_i is the concatenation of its own word embedding $e(w_i)$, the left context embedding $c_l(w_i)$ and the right context embedding $c_r(w_i)$. $c_l(w_i)$ and $c_r(w_i)$ are the hidden states of bidirectional LSTM corresponding to word w_i. In this manner, using the contextual information, RCNN is better able to disambiguate the meaning of the word w_i compared to conventional neural models that only use a fixed window. Embedding vector x_i is then passed through a linear layer that maps this long concatenated encoding vector to a new representation which captures the most useful semantics for representing the text. In the next step, max pooling is applied across all sequences of text. This converts varying length text into a fixed dimension vector and finally, we map this to the output layer. This can be summarized using the following equations:

$$x_i = [c_l(w_i); e(w_i); c_r(w_i)]$$

$$y_i = tanh(Wx_i + b)$$

$$y = \max_{i=1}^{n} y_i$$

$\max(.)$ is an element-wise function. The k-th element of y is the maximum of the k-th elements of y_i.

IV. Experiments and Results

A. Dataset

The dataset contains job descriptions of varied job profiles and corresponding O*NET code. The job descriptions summarize the responsibilities, qualifications and skills for the job. It also includes important company details such as company mission and culture. The dataset covers 480 O*NET categories of the total 1110 detailed categories. For each O*NET class, the dataset contains 1500 job descriptions, where we randomly sampled 1000 jobs for training and 500 jobs for testing.

B. O*NET coding using Job Title

Previous work on Occupation Coding utilize job title to find the occupation category which contains relatively less information compared to complete job description. To validate our assertion that using full job description improves classification accuracy, it is important that we compare our work with previous approaches. However, most of the previous work focuses on Occupation Coding systems other than O*NET. Therefore, they cannot be used for comparison. Carotene [4] is the only automatic Occupation Coding software, that codes job titles to O*NET categories using supervised machine learning algorithms. However, we were unable to find an open-source implementation of the same. Therefore, for comparison, we train LSTM, CNN and RCNN only job title instead of the complete job description, to find O*NET category. We use these model since they have achieved better results in comparison to previous machine learning algorithms such as SVM and KNN used by Carotene on a variety of Natural Language tasks.

C. Implementation Details

We use Spacy [17] for word and sentence tokenization. To obtain word embeddings we used fastText model [18] trained on Wikipedia 2017, UMBC webbase corpus and statmt.org news dataset. For the CNN model, we use three different filter sizes (2, 3 and 4) to perform convolution. The number of output channels after convolution operation is set to 100. We use ReLU for non-linear activation. For the LSTM model, the size of the hidden layer was set to 100. Since we use a bidirectional LSTM, the final embedding size equals to 200. Similarly, for RCNN, we use bidirectional LSTM network with a single recurrent layer and set the hidden layer size to 100. Tanh is used as an activation function. Further, in the HAN network, we set the GRU cells, with number of neurons in every hidden state as 100 for both sentence encoding and document encoding. Since we use a bidirectional GRU, the final document embedding size becomes 200. Tanh is used as the non-linear activation throughout the HAN network. Adam optimizer is used to train all models with learning rate set to 1e-3. Dropout of 0.5 is used for regularization. The models are implemented using Pytorch and run on a system with NVIDIA GeForce GTX 1080ti GPU and 64GB RAM.

D. Results and Discussion

From Table I we see that, in classifying job titles to O*NET categories, RCNN performs better than LSTM and CNN models. Additionally, we see significant improvement in predicting O*NET categories using job description and job title in comparison to only using the later. The additional information allows to better distinguish similar O*NET classes. Also, the HAN models outperform LSTM, CNN and RCNN emphasising the usefulness of attention based hierarchical text modelling. Lastly, the performance of the HAN model is higher than that of the HAN with hierarchical output layers even though the later utilizes the hierarchical structure of O*NET codes in model training. This can be explained from Figure 1 which allows for visualization of O*NET categories.

For each O*NET class, we randomly sample 10 job descriptions and compute their vector representations using the trained HAN model. We then take the average of these vector representations and the resultant vector is used to characterize the O*NET categories. Finally, to visualize these vectors we use t-SNE to project them into a 2D space. In the figure, O*NET categories belonging to the same major group are marked using the same shape and colour. Whereas, O*NET categories belonging to different major groups have different shape and colour.

From the figure, we see that O*NET categories belonging to the same major group do not cluster together. Further, there are many O*NET categories with the nearest neighbour belonging to different major group than itself. For instance, O*NET

TABLE I
ACCURACY COMPARISON FOR VARIOUS OCCUPATION CODING MODELS (%)

	Level 4	Level 3	Level 2	Level 1
LSTM (job titile)	79.9	82.1	86.0	89.4
CNN (job titile)	80.1	82.1	86.2	89.7
RCNN (job titile)	80.3	82.4	86.4	89.8
LSTM	83.7	85.6	88.6	92.5
CNN	84.3	86.1	89.3	93.1
RCNN	84.6	86.9	89.8	93.6
HAN	**87.7**	**89.2**	**91.7**	**94.1**
HAN + Hierarchy	86.2	87.1	89.7	92.2

Level 4: O*NET Occupations; Level 3: Broad Occupations; Level 2: Minor Group Occupations; Level 1: Major Group Occupations. (job titile) indicates that the corresponding model is trained only using the job tile.

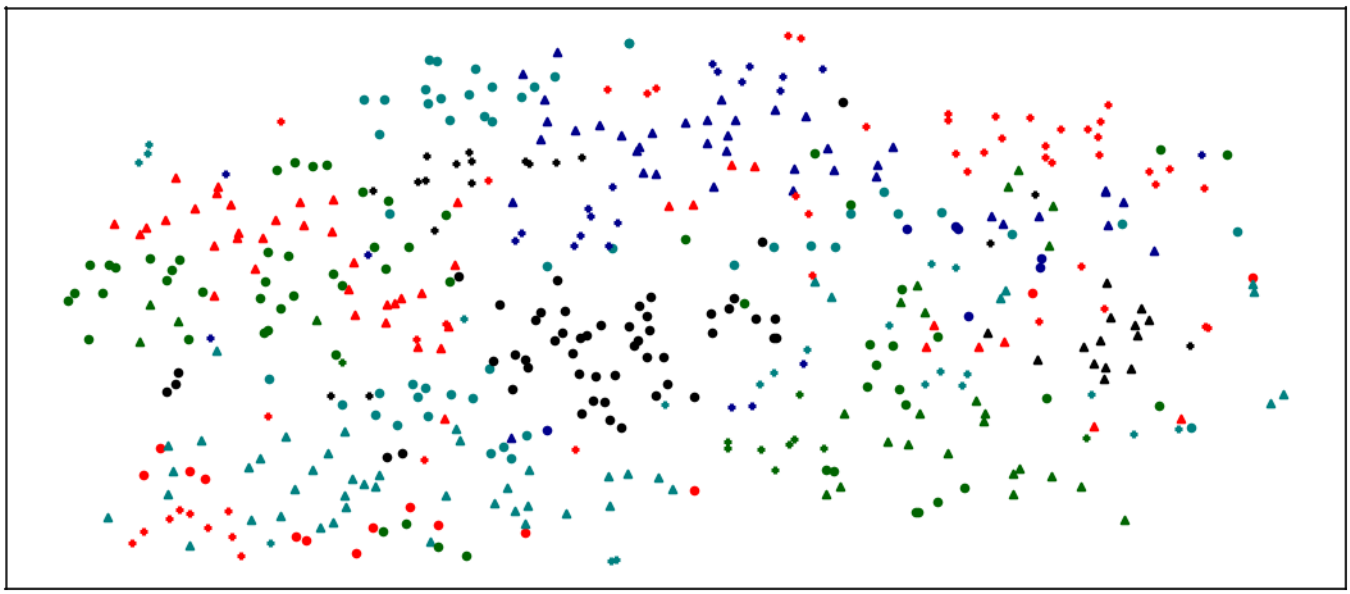

Fig. 1. t-SNE visualization of O*NET categories using HAN model.

category 43-6012.00 (Legal Secretaries) and 23-2011.00 (Paralegals and Legal Assistants) belong to different major groups. Further, 43-9022.00 (Word Processors and Typists) belong to the same major group. However, 43-6012.00 and 23-2011.00 share more similarities compared to 43-6012.00 and 43-9022.00, as the main task in both 43-6012.00 and 23-2011.00 is to prepare and process legal documents. Similarly, the O*NET code 17-3013.00 (Mechanical Drafters) is more similar to 27-1021.00 (Commercial and Industrial Designers) than to 17-3024.00 (Electro-Mechanical Technicians) even though the latter belongs to the same major group. Figure 1 shows that there are many more such cases. Hence when attempting to incorporate hierarchy into model training, dissimilar O*NET codes are brought closer, while similar O*NET categories are pushed farther apart. This has an adverse effect on model learning.

V. CONCLUSION

In this paper, we implemented deep learning models to automatically classify job descriptions to O*NET categories with high accuracy. Experimental results show that the HAN model outperforms other approaches. The hierarchical text modelling and attention mechanism allows HAN to better handle large size of job description and similar O*NET classes in comparison to other approaches. Further, in O*NET taxonomy, many pairs of classes belonging to different major group shared more similarities than pairs of classes belonging to the same major group. We saw that incorporating class hierarchy with such characteristics adversely affects model performance.

REFERENCES

[1] Y. Jung, J. Yoo, S.-H. Myaeng, and D.-C. Han, "A web-based automated system for industry and occupation coding," in *International Conference*

on Web Information Systems Engineering. Springer, 2008, pp. 443–457.

[2] H. Gweon, M. Schonlau, L. Kaczmirek, M. Blohm, and S. Steiner, "Improving the accuracy of automated occupation coding at any production rate," *Available at SSRN 2777765*, 2016.

[3] G. Kirby, J. Carson, F. Dunlop, C. Dibben, A. Dearle, L. Williamson, E. Garrett, and A. Reid, "Automatic methods for coding historical occupation descriptions to standard classifications," in *Population Reconstruction*. Springer, 2015, pp. 43–60.

[4] F. Javed, Q. Luo, M. McNair, F. Jacob, M. Zhao, and T. S. Kang, "Carotene: A job title classification system for the online recruitment domain," in *2015 IEEE First International Conference on Big Data Computing Service and Applications*. IEEE, 2015, pp. 286–293.

[5] M. Schierholz, "Automating survey coding for occupation," Ph.D. dissertation, Ludwig-Maximilians-Universitat Munchen, 2014.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[8] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[9] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, 2016, pp. 1480–1489.

[10] C.-A. Brust and J. Denzler, "Integrating domain knowledge: using hierarchies to improve deep classifiers," *arXiv preprint arXiv:1811.07125*, 2018.

[11] Z. Yan, H. Zhang, R. Piramuthu, V. Jagadeesh, D. DeCoste, W. Di, and Y. Yu, "Hd-cnn: hierarchical deep convolutional neural networks for large scale visual recognition," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2740–2748.

[12] T. Xiong and P. Manggala, "Hierarchical classification with hierarchical attention networks," *KDD Deep Learning Day*, 2018.

[13] B.-C. Chen, R. H. Creecy, and M. V. Appel, "Occupation coding'," *Journal of Official Statistics*, vol. 9, no. 4, pp. 729–745, 1993.

[14] M. D. Patel, K. M. Rose, C. R. Owens, H. Bang, and J. S. Kaufman, "Performance of automated and manual coding systems for occupational data: a case study of historical records," *American journal of industrial medicine*, vol. 55, no. 3, pp. 228–231, 2012.

[15] R. Jones and P. Elias, "Cascot: Computer-assisted structured coding tool," *Coventry: Warwick Institute for Employment Research, University of Warwick*, 2004.

[16] E. Rowe, C. Wong, and A. C. Staff, *An Introduction to the ACTR Coding System*. Bureau of the Census, 1994.

[17] M. Honnibal and I. Montani, "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing," 2017, to appear.

[18] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in pre-training distributed word representations," in *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[19] D. E. Russ, K.-Y. Ho, J. S. Colt, K. R. Armenti, D. Baris, W.-H. Chow, F. Davis, A. Johnson, M. P. Purdue, M. R. Karagas *et al.*, "Computer-based coding of free-text job descriptions to efficiently identify occupations in epidemiological studies," *Occup Environ Med*, vol. 73, no. 6, pp. 417–424, 2016.

[20] K. Takahashi, H. Takamura, and M. Okumura, "Automatic occupation coding with combination of machine learning and hand-crafted rules," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2005, pp. 269–279.

[21] M. Thompson, M. E. Kornbau, and J. Vesely, "Creating an automated industry and occupation coding process for the american community survey," *unpublished report*, 2012.

[22] J. Y. Tourigny and J. Moloney, *The 1991 Canadian Census of Population experience with automated coding*. Statistics Canada, 1997.

[23] M. Wenzowski, "Actr–a generalised automated coding system," *Survey Methodology*, vol. 14, no. 2, pp. 299–308, 1988.

[24] D. Laurison and S. Friedman, "The class pay gap in higher professional and managerial occupations," *American Sociological Review*, vol. 81, no. 4, pp. 668–695, 2016.

[25] C. N. Silla and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, pp. 31–72, 2011.

[26] T. Koeman, N. S. Offermans, Y. Christopher-de Vries, P. Slottje, P. A. Van Den Brandt, R. A. Goldbohm, H. Kromhout, and R. Vermeulen, "Jems and incompatible occupational coding systems: effect of manual and automatic recoding of job codes on exposure assignment," *Annals of occupational hygiene*, vol. 57, no. 1, pp. 107–114, 2012.

[27] N. Nahoomi, "Automatically coding occupation titles to a standard occupation classification," Ph.D. dissertation, The University of Guelph, 2018.