HR Automation: NER For Resumes and Job Matching

Adeel Subhan Fateh Muhammad Fahad Baig
adeelisatwork@gmail.com fatehmuhammad2002@gmail.com fahadbaig221@gmail.com
National University of Computing and Emerging Sciences, Islamabad

Abstract

This paper intends to automate the hiring process by filtering out necessary information from CVs and by giving similarity index between job description and CV. We intend to solve this problem using Natural Language Processing and Deep Learning techniques. We have used NER(Named Entity Recognition) [1] for extracting required information(Skills, Institute, Experience, Contact info, Email address, Location) and Cosine Similarity to find relevance between Job description and Resumes.

Keywords Named Entity Recognition, Resume Parsing, Cosine Similarity, TF-IDF, Training, Deep Learning, Machine Learning, Summarizing, Human Resources, Automation, Natural Language Processing.

1. Problem statement

As the world population increases, the need for jobs also increases side by side. The HR(Human Resource) department has a lot of work on its shoulders in the hiring process. Differentiating between individuals and looking at their CVs is a tedious task to do. A lot of time and resources are required to extract required information from CVs if it's done manually.

The way we used to hire employees has drastically changed over the years. In the 80s companies would put out their job description on newspapers and radios and the interested applicants would contact them with post or reach out them physically. From there on the their applications would be sorted, read, and post was sent back if the company was interested. This was a very laborious and tiring task back then. With the introduction of web, physical mails were replaced by emails and newspapers with online web platforms. Although the reach of people extended by using mails and online hiring platforms, so did the population and applications, meaning time to sort out through resumes and select right candidates got even more difficult. If resumes are somehow parsed and important information is extracted from them which also tells if the person is fit for the job description, more than 80 percent of work is already done for the hiring department.

2. Introduction

2.1 Details

Recent advancements in computing power and machine learning techniques have enabled extraction of useful data from unstructured text possible which was a very challenging task few years a go. We make use of labeled data to train models which can do range of tasks from autonomous driving to recommendation systems. As we progress forward we benefit from previous work (libraries/annotators/data) that makes doing tasks much easier.

The specific problem we're focusing on is to enable automation in hiring and selecting candidates which have the right skill that match with job description of employer. The solution to this is to extract key entities(which matter most to a employer).For job description and resume match we're going to use cosine similarity which is a measure to find similarity between two documents irrespective of their size.

The biggest challenge in training NLP models is to have a very high quality labelled data set which is relevant to its real world test case.Upon looking on the web, there wasn't any good labelled data set(some had overlapping labels, other were present in outdated format), so we had to annotate data by ourselves.

In this paper we solved problem of train RESUME NER model by using spaCy NER model [2], one of the most advanced NER model, along with Sklearn to use cosine similarity to find relevance between resume and job description.

2.2 Motivation

The world is moving towards rapid technological changes day by day. The need for betterment is expected in every aspect of life. The same is the case here by making the work easier for hiring managers of different companies. This project will gather the information of candidates and will also filter them out based on education, experience and their motivation to join the company. This project can be beneficial not only for companies but for different government institutions/organizations in their hiring process. Using NER and Cosine Similarity vector we can find extract useful entities and match percentage respectively. The range of applying NER models is very wide and can identify information in different format irrespective of their position.

2.3 Background

2.3.1 spaCy

A named entity is a word or a phrase that clearly identifies one item from a set of other items that have similar attributes.[17]. NER can identify: Name, Places, Cars, Locations, or anything you can train it on. It has a wide variety of applications from identifying medicines to parsing resumes.

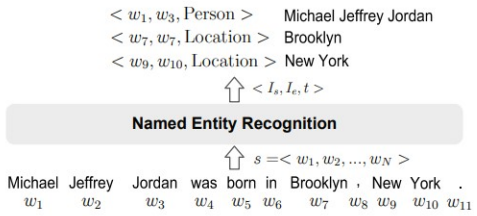
Figure 1. Statistics of spaCy and other NER models.

	SpaCy's	Standford	TensorFlow	OpenNLP
Training accuracy	100%	99.5%	99%	99%
Training loss	0.00000001029	0.00000002137	0.0229	0.00000142
F1-score	100%	94%	97%	96.5%
Prediction probability	100%	90%	96%	98.3%

2.3.2 NER

A named entity is a word or a phrase that clearly identifies one item from a set of other items that have similar attributes.[17]. NER can identify: Name, Places, Cars, Locations, or anything you can train it on. It has a wide variety of applications from identifying medicines to parsing resumes.

Figure 2. An illustration of the named entity recognition task.



3. Related work

The NER work for English started way back in 90s and since then it has made a lot of improvement to the point where it's used by brands, people to identify key elements to their requirements.

In recent years, automatic named entity recognition and extraction systems have become one of the popular research areas that a considerable number of studies have been addressed on developing these systems. Morgan, uses a highly sophisticated linguistic analysis [18], Grishman introduce NYU systems that use handcrafted rules [19]. These approaches are relying on manually coded rules and manually compiled corpora. These kinds of models have better results for restricted domains, are capable of detecting complex entities that learning models have difficulty with. However, the rule-based NE systems lack the ability of portability and robustness, and furthermore the high cost of the rule maintains increases even though the data is slightly changed. These type of approaches are often domain and language specific and do not necessarily adapt well to new domains and languages. In Machine Learning-based NER system, the purpose of Named Entity Recognition approach is converting identification problem into a classification problem and employs a classification statistical model to solve it. In this type of approach, the systems look for patterns and relationships into text to make a model using statistical models and machine learning algorithms. The systems identify and classify nouns into particular classes such as persons, locations, times, etc base on this model, using machine learning algorithms.

4. Your approach

4.1 DATASET PREPARATION

We collected data from multiple sources, mainly from curriculum vitae dataset[3], gathered it into one CSV file followed by these prepossessing steps:

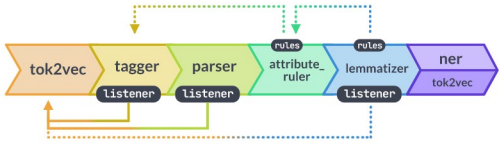
- Normalizing Data
- Converting Unicode characters to ASCII
- Removing unnecessary characters and punctuation.

After Pre-Processing, we moved on to annotation, for which we needed to convert dataset in text format, one resume per line. Once done with preparing text file, we used NER Annotator[4], an open-source software to annotate dataset in spaCy format.Following labels were annotated in training process:

- Name
- Phone Number
- Email Address
- Institute of Higher Education
- Degree Details
- Location
- Skills

4.2 NER Architecture

Figure 3. spaCy NER model architecture



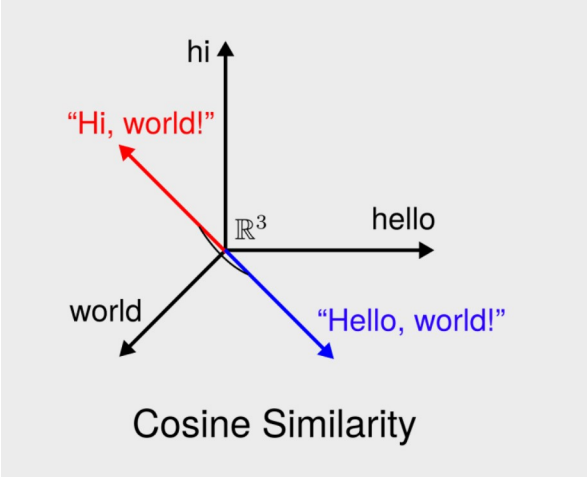
spaCy has a lot of tools which are used to text processing and can be customized. spaCy requires to convert JSON/IOB data to be converted in their custom dataset which acts as input to pipeline. The pipeline elements are modular and can be customized for requires test case, I.e. optional use of GPU, weather to optimize efficiency or accuracy. These can be updated by individual implementation as weights update according to every specific case.

In our case we mainly focused on tok2vec component and NER component. NER is a deep learning model using CNNs, Convolution Neural Network with LSTMs. This form from the basis of transition-based framework. The model used gradient descent and updates the weights of models using back propagation. spaCy train API enables to start from scratch (A blank English model with no previous learned entities) . We can import a pre-trained language model so we have a leverage over few overlapping entities and that's often good if the training data is limited.

4.3 Cosine Similarity

Cosine similarity is used to determine similarity between two words, hence we will use it to find correlation between job description and resume. Firstly, we summarize the resume (to about 30 percent of its original size) using gensim summarizer[12]. To find cosine similarity keywords are determined based on term frequency, TF-IDF. It takes into consideration multi-word expression(MWE) candidates [13] to determine keywords. This research primarily demonstrates implantation of text relevance calculation between resume and job description. Cosine similarity is useful to

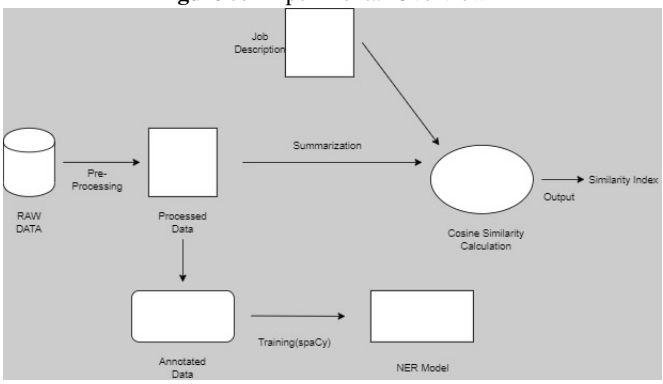
Figure 4. Embedding of Cosine Similarity



measure the similarity between two documents based on terms of their subject matter [14]. Before using cosine similarity, we need to pre-process the text we're using(resume text in this case). Firstly there's removal of punctuation as we do not want them to have any affect on semantics of word. Secondly we need to convert all words to lower case so vectors can have same value. Lastly, tokenizing is done on these words where every token is compared to stop words. This is followed by finding root word(also called stemming[15]).

5. Evaluation and Experiments

Figure 5. Experimental Overview



5.1 Experimental Details

The tok2vec is customizable and is able to provide modifiable vectors. When working with NER in spaCy, there is Work2vec embedding[5] or Glove embedding[6], they provide dynamic and static embedding which are really embedding of meaning and context of word. If you want to go to dynamic route you can chose ELMO[7] and BERT[8].To achieve dynamic embedding we need to consider our training on different language models. spaCy has achieved this by using deep learning model with large dataset in specific context by Pretrain wrapper API [9]. Once training is done the output is embedding model relevant to domain. We experimented with different configurations to achieve optimal performance and accuracy using CPU. We tested our experiments on Google Colab [10], using spaCy v3 (3.3.0) [11] with Python 3.7.13 environment running on Intel Xeon with 8 GB Ram.

5.1.1 Evaluation Metrics

Evaluation is needed for verification of model to check its performance and find the best possible configurations(hyperparameters) for it. We measure the performance of our NER model using F-measure, Recall and Precision.

Precision means the percentage of your results which are relevant. On the other hand, recall refers to the percentage of total relevant results correctly classified by your algorithm.[21]
F1 score is defined as the harmonic mean between precision and recall. It is used as a statistical measure to rate performance. In other words, an F1-score (from 0 to 9, 0 being lowest and 9 being the highest) is a mean of an individual's performance, based on two factors i.e. precision and recall.[22]
Accuracy is just ratio of correctly classified tests to total tests.

Figure 6. Formulas for accuracy,precision,recall and f1

F1-Calcuation Formula:

$$F1 - measure = 2\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Recall Formula:

$$\text{Recall} = \frac{TP}{TP+FP}$$

Precision Formula:

$$\text{Precision} = \frac{TP}{TP+FN}$$

Accuracy Formula:

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN}$$

5.2 Tests

While training we tried multiple configurations and these are results from out trials. Results may vary depending on your hardware configurations and version of python and spaCy.

5.2.1 Using Blank spaCy model

To start off, we trained on blank English model with no pre-trained entities what so ever, used the annotated data to train model on our custom entities(Name, Location, Skills, Degree, etc.) .We did not provide any tokens to vector layer and used the default setting on pipeline. Initially, we used 30 percent of labelled data and model was performing well on entities which followed similar pattern throughout resumes like name, contact info and email, but it had hard time determining skills and work experience. We trained 3 blank models(30,60,90 percent) with dropout rate (0.2). The performance was marginally better on 60 percent data but training on 90 percent of the data did not show any significant improvements.

5.2.2 Using Pre-trained Model

To check if using spaCy pre-trained packages would help in obtaining accuracy, we did the same training using semantic model. It allowed us to initialize weights of model with custom vector layer in neural and convolution layers. spaCy allows us to load the model by just downloading it. This has enabled implementation to obtain dynamic word embedding using Pre-trained Language model[16] . By using pre-training we trained the custom NER model with same entities and data and here's what we notices: There was significant improvement over blank model to start with, but when dataset increased, blank model and Pre-trained model started to level out with almost the same accuracy.

Table 1. Evaluation Results of each entity

Entity	P	R	F
OBJECTIVE	100.00	100.00	100.00
NAME	100.00	100.00	100.00
DESIGNATION	98.00	99.32	98.66
CONTACT NUMBER	100.00	97.87	98.92
EMAIL ADDRESS	100.00	100.00	100.00
SKILLS	98.81	98.42	98.62
DEGREE	95.06	98.72	96.86
INSTITUTE	98.81	95.40	97.08
COMPANIES WORKED AT	99.05	98.11	98.58
LOCATION	100.00	88.24	93.75

Table 2. Average Results

TOK	100.00
NER P	98.66
NER R	98.29
NER F	98.47
SPEED	2806

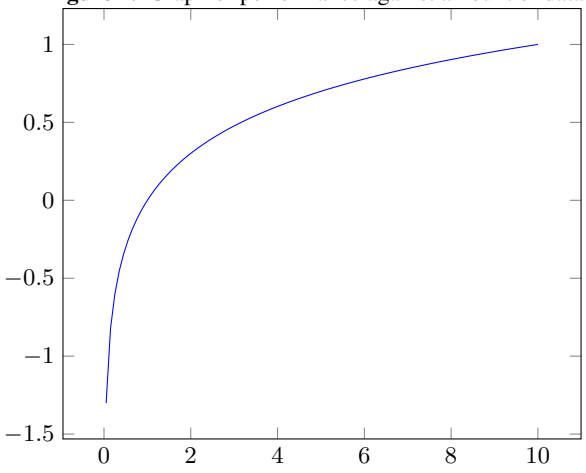
5.2.3 Results

The figure above shows evaluation of model based on test data provided, and to our surprise, the model performed really well while being trained on limited data.

6. Sidenotes

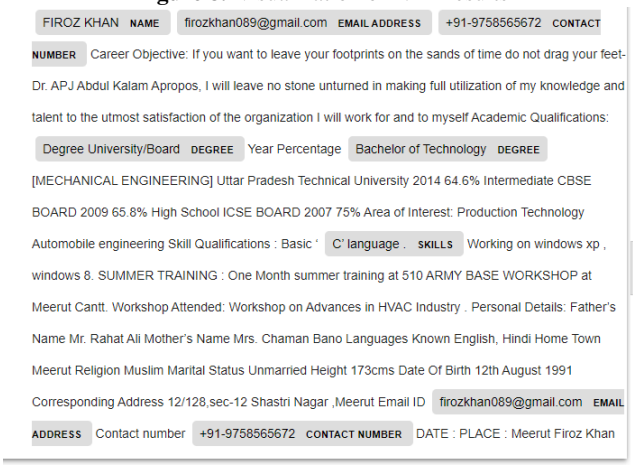
- As seen in the experiments, if dataset is increased the model yields more accurate results on test data and gain is obvious especially if it's a blank model as it's learning entities from scratch with no previous initialization whatsoever. The difference is because pre-trained model uses initialized vectors of relevant domain such as in resume. The performance of model with respect to data is shown in the graph below.

Figure 7. Graph of performance against amount of data



- A key observation of the results presented is that the F1 score of a model trained with our approach with just 80 percent of available training data (0.734) outperforms the F1 score of the blank spaCy model (0.704) trained with 100 percent of the available training data. Clearly, leveraging pre-trained models with partial overlap with the entities provides significant benefits. In future work, we plan to increase the number of entities and experiment with how the number of entities affect performance of the trained models. We also plan to release our pre-trained model with human resource domain entities that can be used for multiple applications. Our approach to the problem using a custom annotation tool and pre-training techniques can be utilized and extended to multiple NLP problems, such as Checking quality of resumes, Homework analysis, Text Summarization etc. The techniques are application domain-agnostic and can be applied to any industrial vertical such as but not limited to: Banking, Insurance, Accounts, Healthcare, Engineering etc., where hiring is required.

Figure 8. visualization of NER results



- According to our tests and experiments, we provided a relatively small training dataset labelled by us, and it performs well on the domain we trained on. This proves the fact that if you have quality labelled data relevant to your domain, it'll yield better results than a large dataset that is poorly labelled/ not relevant to required domain.

References

[1] Raza Ali, Umar Farooq, Umair Arshad, Waseem Shahzad, and Mirza Omer Beg. Hate speech detection on twitter using transfer learning. *Computer Speech & Language*, 74:101365, 2022.

[2] Hamza M Alvi, Hareem Sahar, Abdul A Bangash, and Mirza O Beg. Ensights: A tool for energy aware software development. In *2017 13th International Conference on Emerging Technologies (ICET)*, pages 1–6. IEEE, 2017.

[3] Hamza Mustafa Alvi, Hammad Majeed, Hasan Mujtaba, and Mirza Omer Beg. Mlee: Method level energy estimation—a machine learning approach. *Sustainable Computing: Informatics and Systems*, 32:100594, 2021.

[4] Talha Anwar and Omer Baig. Tac at semeval-2020 task 12: Ensembling approach for multilingual offensive language identification in social media. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 2177–2182, 2020.

[5] Muhammad Umair Arshad, Muhammad Farrukh Bashir, Adil Majeed, Waseem Shahzad, and Mirza Omer Beg. Corpus for emotion detection on roman urdu. In *2019 22nd International Multitopic Conference (INMIC)*, pages 1–6. IEEE, 2019.

[6] Muhammad Asad, Muhammad Asim, Talha Javed, Mirza O Beg, Hasan Mujtaba, and Sohail Abbas. Deepdetect: detection of dis-

tributed denial of service attacks using deep learning. *The Computer Journal*, 63(7):983–994, 2020.

[7] Mubashar Nazar Awan and Mirza Omer Beg. Top-rank: a topicalpostionrank for extraction and classification of keyphrases in text. *Computer Speech & Language*, 65:101–116, 2021.

[8] Abdul Ali Bangash, Hareem Sahar, and Mirza Omer Beg. A methodology for relating software structure with energy consumption. In *2017 IEEE 17th International Working Conference on Source Code Analysis and Manipulation (SCAM)*, pages 111–120. IEEE, 2017.

[9] Muhammad Farrukh Bashir, Abdul Rehman Javed, Muhammad Umair Arshad, Thippa Reddy Gadekallu, Waseem Shahzad, and Mirza Omer Beg. Context aware emotion detection from low resource urdu language using deep neural network. *Transactions on Asian and Low-Resource Language Information Processing*, 2022.

[10] M Beg. Critical path heuristic for automatic parallelization. *University of Waterloo, David R. Cheriton School of Computer Science, Technical Report CS-2008-16*, 2008.

[11] Mirza Beg and Peter van Beek. A constraint programming approach for integrated spatial and temporal scheduling for clustered architectures. *ACM Transactions on Embedded Computing Systems (TECS)*, 13(1):1–23, 2013.

[12] Mirza Beg and Mike Dahlin. A memory accounting interface for the java programming language. *Technical Report CS-TR-01–40, University of Texas at Austin*, 2001.

[13] Mirza Beg and Peter Van Beek. A graph theoretic approach to cache-conscious placement of data for direct mapped caches. In *Proceedings of the 2010 international symposium on Memory management*, pages 113–120, 2010.

[14] Mirza Beg and Peter Van Beek. A constraint programming approach for instruction assignment. In *2011 15th Workshop on Interaction between Compilers and Computer Architectures*, pages 25–34. IEEE, 2011.

[15] Mirza O Beg, Mubashar Nazar Awan, and Syed Shahzaib Ali. Algorithmic machine learning for prediction of stock prices. In *FinTech as a Disruptive Technology for Financial Institutions*, pages 142–169. IGI Global, 2019.

[16] Mirza Omer Beg. Combinatorial problems in compiler optimization. 2013.

[17] Noman Dilawar, Hammad Majeed, Mirza Omer Beg, Naveed Ejaz, Khan Muhammad, Irfan Mehmood, and Yunyoung Nam. Understanding citizen issues through reviews: A step towards data informed planning in smart cities. *Applied Sciences*, 8(9):1589, 2018.

[18] Muhammad Umer Farooq, Mirza Omer Beg, et al. Bigdata analysis of stack overflow for energy consumption of android framework. In *2019 International Conference on Innovative Computing (ICIC)*, pages 1–9. IEEE, 2019.

[19] Muhammad Umer Farooq, Saif Ur Rehman Khan, and Mirza Omer Beg. Melta: A method level energy estimation technique for android development. In *2019 International Conference on Innovative Computing (ICIC)*, pages 1–10. IEEE, 2019.

[20] Sadia Ismail, Hasan Mujtaba, and Mirza Omer Beg. Spems: A sustainable parasitic energy management system for smart homes. *Energy and Buildings*, 252:111429, 2021.

[21] Abdul Rehman Javed, Mirza Omer Beg, Muhammad Asim, Thar Baker, and Ali Hilal Al-Bayatti. Alphalogger: Detecting motion-based side-channel attack using smartphone keystrokes. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–14, 2020.

[22] Abdul Rehman Javed, Muhammad Usman Sarwar, Mirza Omer Beg, Muhammad Asim, Thar Baker, and Hissam Tawfik. A collaborative healthcare framework for shared healthcare plan with ambient intelligence. *Human-centric Computing and Information Sciences*, 10(1):1–21, 2020.

[23] Hafiz Tayyeb Javed, Mirza Omer Beg, Hasan Mujtaba, Hammad Majeed, and Muhammad Asim. Fairness in real-time energy pricing for smart grid using unsupervised learning. *The Computer Journal*, 62(3):414–429, 2019.

[24] Muhammad Saad Javed, Hammad Majeed, Hasan Mujtaba, and Mirza Omer Beg. Fake reviews classification using deep learning ensemble of shallow convolutions. *Journal of Computational Social Science*, 4(2):883–902, 2021.

[25] Martin Karsten, Srinivasan Keshav, Sanjiva Prasad, and Mirza Beg. An axiomatic basis for communication. *ACM SIGCOMM Computer Communication Review*, 37(4):217–228, 2007.

[26] Hussain S Khawaja, Mirza O Beg, and Saira Qamar. Domain specific emotion lexicon expansion. In *2018 14th International Conference on Emerging Technologies (ICET)*, pages 1–5. IEEE, 2018.

[27] Adil Majeed, Hasan Mujtaba, and Mirza Omer Beg. Emotion detection in roman urdu text using machine learning. In *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering Workshops*, pages 125–130, 2020.

[28] Hammad Majeed, Abdul Wali, and Mirza Beg. Optimizing genetic programming by exploiting semantic impact of sub trees. *Swarm and Evolutionary Computation*, 65:100923, 2021.

[29] Bilal Naeem, Aymen Khan, Mirza Omer Beg, and Hasan Mujtaba. A deep learning framework for clickbait detection on social area network using natural language cues. *Journal of Computational Social Science*, pages 1–13, 2020.

[30] Saad Naeem, Majid Iqbal, Muhammad Saqib, Muhammad Saad, Muhammad Soban Raza, Zaid Ali, Naveed Akhtar, Mirza Omer Beg, Waseem Shahzad, and Muhhamad Umair Arshad. Subspace gaussian mixture model for continuous urdu speech recognition using kaldi. In *2020 14th International Conference on Open Source Systems and Technologies (ICOSST)*, pages 1–7. IEEE, 2020.

[31] Saira Qamar, Hasan Mujtaba, Hammad Majeed, and Mirza Omer Beg. Relationship identification between conversational agents using emotion analysis. *Cognitive Computation*, pages 1–15, 2021.

[32] Hareem Sahar, Abdul A Bangash, and Mirza O Beg. Towards energy aware object-oriented development of android applications. *Sustainable Computing: Informatics and Systems*, 21:28–46, 2019.

[33] Muhammad Tariq, Hammad Majeed, Mirza Omer Beg, Farrukh Aslam Khan, and Abdelouahid Derhab. Accurate detection of sitting posture activities in a secure iot based assisted living environment. *Future Generation Computer Systems*, 92:745–757, 2019.

[34] Ahmed Uzair, Mirza O Beg, Hasan Mujtaba, and Hammad Majeed. Weec: Web energy efficient computing: A machine learning approach. *Sustainable Computing: Informatics and Systems*, 22:230–243, 2019.

[35] Adeel Zafar, Hasan Mujtaba, Sohrab Ashiq, and Mirza Omer Beg. A constructive approach for general video game level generation. In *2019 11th Computer Science and Electronic Engineering (CEEC)*, pages 102–107. IEEE, 2019.

[36] Adeel Zafar, Hasan Mujtaba, Mirza Tauseef Baig, and Mirza Omer Beg. Using patterns as objectives for general video game level generation. *ICGA Journal*, 41(2):66–77, 2019.

[37] Adeel Zafar, Hasan Mujtaba, and Mirza Omer Beg. Search-based procedural content generation for gvg-lg. *Applied Soft Computing*, 86:105909, 2020.

[38] Adeel Zafar, Hasan Mujtaba, Mirza Omer Beg, and Sajid Ali. Deceptive level generator. In *AIIDE Workshops*, 2018.

[39] Named Entity Recognition: https://en.wikipedia.org/wiki/Named-entity_recognition.

[40] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, Jeff Dean 2013. Distributed Representations of Words and Phrases and their Compositionality In Advances in Neural Information Processing Systems 26

[41] Jeffrey Pennington, Richard Socher, Christopher Manning 2014 Glove: Global Vectors for Word Representation in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)

[42] Matthew Peters et al. 2018. Deep contextualized word representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistic

[43] Jacob Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding . In Proceedings of the 2019 Annual Conference of the North American Chapterof the Association for Computational Linguistics, NAACL-HLT '19.

[44] Matthew Hannibal; Language Model Pre-training in spaCy.

[45]https://radimrehurek.com/gensim_3.8.3/summarization/summariser.html

[46] Gunawan D, Amalia A, and Charisma I 2016 Automatic extraction of multiword expression candidates for Indonesian language 2016 6th IEEE Int. Conf. on Control System, Computing and Engineering (ICCSCE) pp 304–309

[47] Singhal A 2001 Modern Information Retrieval: A Brief Overview Bulletin of the Technical Committee on Data Engineering pp 35–43

[48] Porter M 2006 "The Porter Stemming Algorithm" Available: https://tartarus.org/martin/PorterStemmer/

[49] Matthew Hannibal; Language Model Pre-training in spaCy. https://spacy.io/usage/v2-1pretraining

[50] R. Sharnagat, "Named entity recognition: A literature survey," Center For Indian Language Technology, 2014.

[51] Morgan, R., and et. al., "University of durham: Description of the LOLITA system as used for MUC-6" In Proc of the MUC-6, NIST, Morgan-Kaufmann Publishers, Columbia, 1995.

[52] R. Grishman, "The NYU System for MUC-6 or Where's the Syntax", In Proceedings of the Sixth Message Understanding Conference (MUC-6), 1995

[53] Furrer, L., Jancso, A., Colic, N., Rinaldi, F. (2019). OGER++: hybrid multi-type entity recognition. Journal of Cheminformatics, 11(1). doi:10.1186/s13321-018-0326-3.