Resumes Named Entity Recognition with Conditional Random Fields

Juan Alejandro Alcántara Minaya
A01703947@exatec.tec.mx

Dr. Benjamín Valdés Aguirre
bvaldes@tec.mx

CONTENTS

I. INTRODUCTION

We are often faced with the debate of humans versus machines, with a lot of people concerned that they will be replaced by robots. However, the goal has always been to strengthen the relationship between automated systems and humans to improve the overall work quality. The objective of this research paper is to provide an automated tool to help Spanish recruiters parse through the large volume of resumes they handle daily.

There are a lot of existing tools that help recruiters in other languages, such as English, to process candidate resumes. Most commonly known as Applicant Tracking Systems (ATS), these systems help filter through a pool of candidates and provide the best match for a job vacancy. However, these tools are not as widespread in the Spanish language. We decided to take on the challenge of creating something similar to help the development of the area.

More significantly, we focused on developing a said tool using a lesser-known type of model called Conditional Random Fields (CRF) since it offers great benefits at low computational costs. Other research has shown how these models compare to traditional deep learning methods and it shows a lot of promise. The biggest challenge, however, is being able to capture and convey the nuances of Spanish resumes so that a model can understand how to process them.

Building tools like this one don't have the objective of displacing people from their jobs, but instead of freeing up their time and effort toward more rewarding tasks. Processing large datasets can be tolling for humans and the best way to combat it is with the help of machines. The future of work is pointed toward collaboration with automated systems and we strive to be part of it.

II. LITERATURE REVIEW

A. Named Entity Recognition (NER)

Named Entity Recognition is a subarea of Natural Language Processing. Its objective is to locate and categorize important words in a text. This process is crucial in applications that focus on information extraction, translation, and overall text analysis. Usually, named entities are classified according to the information they provide. For example, NER tasks frequently focus on extracting people, locations, and organizations from large text corpora [1].

There are two main challenges in named entity recognition tasks. The first one is determining the correct boundaries for a certain entity and the second one is determining the correct class for said entity. The author explains that there are ambiguous examples where it is easy for the models to make a mistake. For instance, the word *Fox* can be interpreted as either a person or an organization. Another challenge arises when you consider that different domains consider different entities essential. For example, in resumes, the word *Python* can refer to a skill, whereas in biology it may be referring to an animal. It is, therefore, crucial to have specifically labeled corpora tailored for the specific domain. [1]

B. Conditional Random Fields (CRF)

Conditional Random Fields are undirected probabilistic graphical models that tackle sequence analysis. It is often confused with Hidden Markov Models (HMMs) since they share some similarities. However, the main difference is that HMMs are a form of generative model and to "define a joint distribution... [it] must enumerate all possible observation sequences - a task which... is intractable unless observation elements are represented as isolated units" [2].

Nonetheless, Conditional Random Fields can model the dependency between different observations without having to increase model complexity. This property also allows the model to consider past and future tokens for the probabilistic prediction of the current token [3].

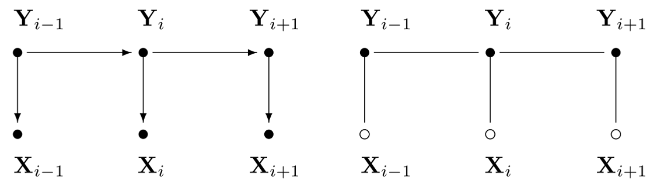

Fig. 1. Graphical structures of simple HMMs (left), and chain-structured case of CRFs (right) for sequences. An open circle indicates that the variable is not generated by the model. [3]

III. Related Work

Resumes fall into the category of semi-structured documents. This characteristic poses a challenge for traditional parsing methods because the contents, although usually grouped by topic, do not share a single format. Computer scientists have had to rely on natural language processing (NLP) techniques to extract the relevant content. Different authors have used part of speech (POS) tagging, *stop word* removal, and stemming to process the resume text content [4], [5], [6].

A. English Resume Parsing

Roy et al. [6] performed another survey of different classifier models such as Random Forests, Multinomial Naive Bayes, Logistic Regression, and Linear Support Vector Machines (LSVM). Their objective was to classify resumes into different job sectors such as sales, consulting, finance, technology, etc. To feed the data into the models they had to remove junk characters, and *stop words* from the resumes. They also had to use stemming and lemmatization. This led to data loss, which is mentioned in the limitations. However, their objective was not to gather the specifics of their resumes, but instead to get a general idea of the candidate profile. Their research showed that the most performant model was the LSVM, with an average accuracy of 78.53%.

This survey showcases that for resume parsing the best model is usually the model that excels at sequence analysis. It shouldn't come as a surprise since resumes are semi-structured documents.

On this note, Ayishathahira et al. [5] focused on effectively gathering named entities by comparing the performance of four models:

- Convolutional Neural Network (CNN)
- Bidirectional Long Short-Term Memory (Bi-LSTM)
- Conditional Random Field (CRF)
- Bi-LSTM and CNN combination (Bi-LSTM-CNN)

They worked with a dataset of 800 resumes sourced privately by one of their sponsors. They removed punctuations and other extraneous characters as well. Nevertheless, the authors didn't use lemmatization or stemming which allowed the sequences to remain mostly intact.

Ayishathahira et. al defined 23 labels to classify information about a candidate's education, occupation, and identification. They trained the models to discover that the CRF implementation yielded the best results. On average, the CRF model had an F-score of 83.70% while the Bi-LSTM-CNN had an F-score of 68.43%. The CRF model beat the Bi-LSTM-CNN model across all the labels, with an average improvement of 15.26%.

The CRF model showed great advantages over its counterparts. It is relatively fast to train, needing less than half an hour to go through the authors' dataset. This speed, combined with its accuracy, makes the CRF model a great candidate to develop effective resume parsing tools.

E. Suhas and E. Manjunath [7] took Ayishathahira's [5] research even further and they were able to develop a mixed model that allowed them to match candidates with job postings. The model is a combination of Named Entity Recognition (NER) and Word2Vec embedding to find the cosine distance between entities in resumes and job postings.

They generated four different iterations of the NER model using, Stanford's NER model as a basis. They either removed more noise or increased the dataset size on each iteration and this translated to an F-score increase from about 52% to more than 80%. The largest increase, of about 15%, was achieved from the third iteration to the fourth iteration by introducing a dictionary of technical skills to the NER and increasing the window size.

Even though this study had a smaller dataset than Ayishathahira's [5], they were able to achieve similar results. Since it is a time-consuming and arduous process to prepare these datasets, this strategy is a great alternative to get the most out of the available resources.

B. Spanish Named Entity Recognition

Based on the research done for English corpus, Copara et. al [8] developed a model that allowed named entity recognition for Spanish text. They decided to evaluate their model by using the CoNLL-2002 (Spanish) dataset since it allows them to compare their performance against the state of the art.

The authors decided to use word embedding and clustering in hopes of achieving better results. They used a large corpus of text both in Spanish and English to generate robust embeddings. Their baseline model achieved an F-Score of 80.02%. After adding embedding and clustering, they were able to increase this score to 82.30%. They noticed that there are words that are more likely to belong to one class than to another class and decided to use distributional prototypes to leverage this pattern. After adding prototypes, the model's F-Score dropped to 81.19%. Nevertheless, they had the hypothesis that some entities share very similar features across languages and decided to use the Brown clusters from English. The model's F-score increased to 82.44%.

Although the study didn't cover resumes specifically, it showcased various strategies to get better performance, even with a Spanish dataset. The most impressive observation is that the CRF model performed very similarly to the current state-of-the-art Deep Learning models.

IV. Problem Statement

According to the Economic Commission for Latin America and the Caribbean (ECLAC), in 2019, an estimated average of 3.8 million workers searched for work daily [9]. The same report estimated a daily average of 1.4 million job offer calls [9]. The supply is almost three times the demand, and it presents itself as an unmanageable task for recruiters across the region. Finding the best employee for a position is time-consuming since it requires combing through many candidates, and often missing essential information.

The problem is that while in English-speaking countries there are tools to aid companies to handle the sheer volume of applications, there is not enough research to create these tools for Spanish-speaking countries.

V. Solution

The most viable solution to this problem is using Conditional Random Fields (CRF) to extract the essential information from candidates' resumes. Compared to the training time and complexity of deep learning methods, CRFs are quicker and simpler to implement. The key to building a robust CRF is choosing the right features and having a large enough dataset. In comparison, Conditional Random Fields are more flexible allowing the model to find relationships between sequences even if they are not immediately next to each other. These characteristics and the research so far suggest that these models will yield good results when parsing resumes, regardless of the language.

It is difficult to settle on a fixed sequence size to analyze each step of the CRF training. Depending on the field, the candidate, and the template, sequences for the same data can vary greatly. Traditionally, other researchers have used a word per word basis to train their models [5], [7]. Having settled on this strategy, we built three different feature sets (see I, II, and III) to train the CRF.

We built the solution using the Python library, `sklearn-crfsuite` which is a `scikit-learn` compatible wrapper for the C++ library `CRFSuite` [10]. The library allows its users to create linear-chain CRFs using multiple training methods such as Limited-memory BFGS (L-BFGS), Stochastic Gradient Descent (SDG), Adaptive Regularization Of Weight Vector (AROW), etc. For this solution, we went with L-BFGS since it allowed us to set L1 and L2 normalization parameters.

After having designed the model, we added preprocessing steps to feed it data. We parsed each resume into word tokens, eliminating common punctuation points such as periods, commas, question marks, etc. The next step was to calculate the features for each word in the resume as described in the tables (I, II, III). It is important to note that for the first and last word of each sequence features belonging to the previous and next word, respectively, were omitted.

CRFs are great to predict what is the most likely transition from one tag to another, which is why they are an attractive solution to analyze semi-structured documents such as resumes. Based on this fact and the existing research so far, we believe this is a sufficient solution to address the problem at hand.

Feature	**Description**
-1:word.istitle	Whether the previous word starts with a capital letter.
-1:word.isdigit	Whether the previous word is an integer.
-1:word.is_year	Whether the previous word is an integer between 1900 and 2100 (inclusive).
-1:word.is_abbr_month	Whether the previous word is a month's name abbreviation.
-1:word.is_full_month	Whether the previous word is a month's name.
-1:word.syllable_count	How many syllables does the previous word have, based on its vowels.
-1:word.lower	The full previous word in lowercase.
-1:word.bigram	The previous word concatenated with the current word.
word.istitle	Whether the current word starts with a capital letter.
word.isdigit	Whether the current word is an integer.
word.is_year	Whether the current word is an integer between 1900 and 2100 (inclusive).
word.len	Then length of the current word
word.is_abbr_month	Whether the current word is a month's name abbreviation.
word.is_full_month	Whether the current word is a month's name.
word.syllable_count	How many syllables does the current word have, based on its vowels.
word.has_at	Whether the current word has an @ symbol.
+1:word.istitle	Whether the next word starts with a capital letter.
+1:word.isdigit	Whether the next word is an integer.
+1:word.is_year	Whether the current word is an integer between 1900 and 2100 (inclusive).
+1:word.is_abbr_month	Whether the next word is a month's name abbreviation.
+1:word.is_full_month	Whether the next word is a month's name.
+1:word.syllable_count	How many syllables does the current word have, based on its vowels.
+1:word.lower	The full next word in lowercase.
+1:word.bigram	The current word concatenated with the next word.

TABLE I
Baseline Features

Feature	**Description**
-1:word.istitle	Whether the previous word starts with a capital letter.
-1:word.isdigit	Whether the previous word is an integer.
-1:word.is_year	Whether the previous word is an integer between 1900 and 2100 (inclusive).
-1:word.is_abbr_month	Whether the previous word is a month's name abbreviation.
-1:word.is_full_month	Whether the previous word is a month's name.
-1:word.syllable_count	How many syllables does the previous word have, based on its vowels.
-1:word.lower	The full previous word in lowercase.
-1:word.bigram	The previous word concatenated with the current word.
-1:word.first_three	The previous word first three characters
-1:word.last_three	The previous word last three characters.
word.istitle	Whether the current word starts with a capital letter.
word.isdigit	Whether the current word is an integer.
word.is_year	Whether the current word is an integer between 1900 and 2100 (inclusive).
word.len	Then length of the current word
word.is_abbr_month	Whether the current word is a month's name abbreviation.
word.is_full_month	Whether the current word is a month's name.
word.syllable_count	How many syllables does the current word have, based on its vowels.
word.has_at	Whether the current word has an @ symbol.
word.first_three	The current word first three characters
word.last_three	The current word last three characters.
+1:word.istitle	Whether the next word starts with a capital letter.
+1:word.isdigit	Whether the next word is an integer.
+1:word.is_year	Whether the current word is an integer between 1900 and 2100 (inclusive).
+1:word.is_abbr_month	Whether the next word is a month's name abbreviation.
+1:word.is_full_month	Whether the next word is a month's name.
+1:word.syllable_count	How many syllables does the current word have, based on its vowels.
+1:word.lower	The full next word in lowercase.
+1:word.bigram	The current word concatenated with the next word.
+1:word.first_three	The next word first three characters
+1:word.last_three	The next word last three characters.

TABLE II
HEAD + TAIL
(Changes marked in yellow)

Feature	**Description**
-1:word.istitle	Whether the previous word starts with a capital letter.
-1:word.isdigit	Whether the previous word is an integer.
-1:word.is_year	Whether the previous word is an integer between 1900 and 2100 (inclusive).
-1:word.is_abbr_month	Whether the previous word is a month's name abbreviation.
-1:word.is_full_month	Whether the previous word is a month's name.
-1:word.syllable_count	How many syllables does the previous word have, based on its vowels.
-1:word.lower	The full previous word in lowercase.
-1:word.bigram	The previous word concatenated with the current word.
-1:word.first_three	The previous word first three characters
-1:word.last_three	The previous word last three characters.
word.istitle	Whether the current word starts with a capital letter.
word.isdigit	Whether the current word is an integer.
word.is_year	Whether the current word is an integer between 1900 and 2100 (inclusive).
word.len	Then length of the current word
word.is_abbr_month	Whether the current word is a month's name abbreviation.
word.is_full_month	Whether the current word is a month's name.
word.syllable_count	How many syllables does the current word have, based on its vowels.
word.has_at	Whether the current word has an @ symbol.
word.first_three	The current word first three characters
word.last_three	The current word last three characters.
word.lower	The full current word in lowercase.
+1:word.istitle	Whether the next word starts with a capital letter.
+1:word.isdigit	Whether the next word is an integer.
+1:word.is_year	Whether the current word is an integer between 1900 and 2100 (inclusive).
+1:word.is_abbr_month	Whether the next word is a month's name abbreviation.
+1:word.is_full_month	Whether the next word is a month's name.
+1:word.syllable_count	How many syllables does the current word have, based on its vowels.
+1:word.lower	The full next word in lowercase.
+1:word.bigram	The current word concatenated with the next word.
+1:word.first_three	The next word first three characters
+1:word.last_three	The next word last three characters.

TABLE III
WHOLE WORD
(Changes marked in yellow)

VI. Experimentation & Evaluation

There is a lack of public Spanish datasets with tagged resumes, which means that we had to gather and build our own. First, we needed to collect Spanish resumes and convert them from their original formats to plain text. This can be challenging due to the great diversity of file formats and resume templates. Building a named entity recognition dataset is a laborious task since it means going through each data sample and labeling the sequences by hand. We were able to accomplish this task thanks to the open-source program, Doccano. Doccano provides a graphical interface to label the documents by selecting the text and assigning a label. After labeling each resume, we used the dataset to train the model and evaluate its performance.

We collected resumes from two main sources: an HR representative at an automotive manufacturing company and Mexico's government representatives' public available resumes. In total, we collected over 90 resumes, of which we could only use 51 due to errors in the text conversion. The resumes comprised different profiles such as engineers, marketers, lawyers, educators, and public servants. Although it would have been ideal to have a larger dataset, we were limited by the available time and resources.

The dataset was split in a 1:5 ratio for testing and training respectively. This proportion was chosen based on the Pareto Principle and it recurrent relevance in different datasets. It is also a commonly used distribution in computer science applications [11].

To make the most out of the available data, we decided to use K-fold cross-validation to optimize the model's hyper-parameters (L1 and L2 normalization). We used `scikit-learn` Randomized search with 100 iterations of 5-fold cross-validation and selected the best model based on the weighted f-score [12].

Since this is a classification problem, we can use different metrics to measure the performance of our model. We can obtain the precision, recall, and f-score by using a confusion matrix. Precision allows us to gauge how often our model correctly predicts the tag for a text sequence. Recall lets us measure how good are the model predictions for the correct class out of all the examples available for said class. F-score will let us relate these previous variables to get a better sense of the robustness of our model. These metrics are popularly used to evaluate classification models and more specifically, these are metrics that have been used in the related work mentioned previously.

It is important to note that recall is often disregarded in computer linguistics since false positives and false negatives are not as influential as in other fields. Although it serves to paint a general picture of a model's performance, it depends on what is the cost of false predictions. On the other hand, precision is highly regarded as a key metric in machine learning and computational linguistics with the caveat that it doesn't reflect correctly how the model handles false predictions. As a result, the f-score is used as a compromise between these two metrics. The F-measure relates the precision and recall by obtaining their harmonic mean [13]. This measure is also used as an evaluation metric for other studies and will allow us to compare the performance of the solution to the state of the art.

VII. Results

After training the model multiple times we obtained thought-provoking results. There are some shared trends across feature sets such as the DATE label not having enough samples to achieve significance. Similarly, labels for very specific entities such as FIRST_NAME, LAST_NAME, and LOCATION have very low recall scores. However, entities with more generic and standardized formats such as EMAIL and PHONE_NUMBER have higher recall scores. These observations are directly linked to the value of the f1-scores, where the lower the recall, the lower the f1-score.

Consequently, the number of samples available for each label affects greatly its performance. The most obvious example of this is when we compare PHONE_NUMBER and uncategorized (NAN) labels. Even though phone numbers have a predictable format, the uncategorized text has a better performance across the board. It becomes evident that when dealing with CRFs we are playing a numbers game rather than a feature engineering game. That's not to say that features don't play a role in these models.

When we look at the comparison across different feature sets (Figure 2), we notice that when the amount of samples doesn't vary significantly, features play a large role in performance improvement. The second feature set, Head + Tail, consistently beats the other models in most of the categories. This is even more palpable when we compare the average f1-score where it beats the other models with a value of 41%. It is important to note that the performance dropped by one decimal point when we added the whole word features. Although more exploration is needed, this may be due to either the difference in total samples between the two feature sets or slight overfitting.

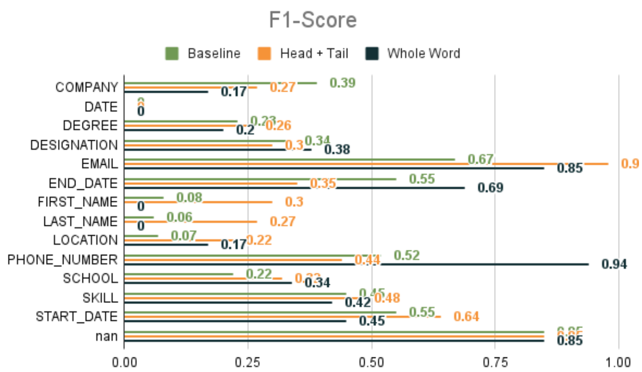

Fig. 2. Comparison of f1-score values for each label across the different feature sets.

In table VII, we can see a comparison between previously done research and our solution. In contrast, the current solution cannot boast great performance, but this is due to

	Precision	Recall	F1-score	Support
COMPANY	0.55	0.30	0.39	481
DEGREE	0.59	0.14	0.23	365
DESIGNATION	0.82	0.21	0.34	373
EMAIL	1.00	0.50	0.67	10
END_DATE	0.87	0.41	0.55	185
FIRST_NAME	0.20	0.05	0.08	19
LAST_NAME	0.33	0.03	0.06	29
LOCATION	0.62	0.04	0.07	134
PHONE_NUMBER	0.86	0.38	0.52	16
SCHOOL	0.28	0.19	0.22	203
SKILL	0.67	0.34	0.45	436
START_DATE	0.75	0.43	0.55	175
nan	0.76	0.96	0.85	5836
Micro Avg.	0.75	0.75	0.75	8262
Macro Avg.	0.64	0.31	0.38	8262
Weighted Avg.	0.73	0.75	0.70	8262

TABLE IV

BASELINE FEATURES RESULTS

	Precision	Recall	F1-score	Support
COMPANY	0.47	0.19	0.27	542
DATE	0.00	0.00	0.00	98
DEGREE	0.51	0.18	0.26	462
DESIGNATION	0.71	0.19	0.30	366
EMAIL	1.00	0.95	0.98	22
END_DATE	0.76	0.23	0.35	247
FIRST_NAME	1.00	0.18	0.30	17
LAST_NAME	1.00	0.15	0.27	26
LOCATION	0.53	0.14	0.22	217
PHONE_NUMBER	0.86	0.30	0.44	20
SCHOOL	0.45	0.25	0.32	365
SKILL	0.59	0.41	0.48	354
START_DATE	0.89	0.50	0.64	207
nan	0.77	0.95	0.85	7318
Accuracy	-	-	0.75	10261
Macro Avg.	0.68	0.33	0.41	10261
Weighted Avg.	0.72	0.75	0.70	10261

TABLE V

HEAD+TAIL FEATURES RESULTS

factors such as the dataset size and feature engineering. Even though the f1-score of the solution is half of the other works, it is important to remember that the model has a decent performance when it comes to a binary classification of essential and nonessential information. Across all the feature sets, the model achieved an 85% f1-score for uncategorized data.

	Precision	Recall	F1-score	Support
COMPANY	0.39	0.11	0.17	405
DATE	0.00	0.00	0.00	77
DEGREE	0.24	0.17	0.20	344
DESIGNATION	0.38	0.38	0.38	261
EMAIL	0.92	0.79	0.85	14
END_DATE	0.91	0.55	0.69	184
FIRST_NAME	0.00	0.00	0.00	19
LAST_NAME	0.00	0.00	0.00	27
LOCATION	0.23	0.13	0.17	255
PHONE_NUMBER	0.81	0.94	0.87	18
SCHOOL	0.48	0.26	0.34	360
SKILL	0.58	0.33	0.42	529
START_DATE	0.65	0.45	0.53	161
nan	0.78	0.92	0.85	6421
Accuracy	-	-	0.73	9075
Macro Avg.	0.45	0.36	0.39	9075
Weighted Avg.	0.68	0.73	0.69	9075

TABLE VI

WHOLE WORD FEATURES RESULTS

Study/Research	**Avg. F1-score**
Ayishathahira et. al (2018) [5]	83.7%
E, Suhas & E, Manjunath (2020) [7]	81%
Copara et. al (2016) [8]	82.44%
Current solution	**41%**

TABLE VII

WHOLE WORD FEATURES RESULTS

VIII. CONCLUSIONS

Natural Language Processing is one of the most challenging tasks when it comes to artificial intelligence. However, that doesn't mean that there is no way to surmount these challenges. With a large enough dataset, it is easier for the model to detect certain patterns in our language. Alternatively, this kind of model benefits from access to a specific database for skills, names, and other proper nouns. Even without these benefits, this CRF is a great fundamental step in the right direction for processing resumes automatically.

The amount of quality data available is one of the most important parts of training any kind of model. In previous works, we have seen the best performance when the most amount of data is available. For instance, Ayishathahira et. al [5] had a dataset of 800 curated resumes and they were able to beat traditional deep learning methods with their CRF model. Even when we look at the different feature sets, the one with the best performance is also the one with the most amount of samples. This project can be taken a step further by collecting a large Spanish resume dataset. This would allow the CRF model to better understand certain nuances of our natural language.

It would have been beneficial to have auxiliary datasets with common instances of certain entities. E, Suhas E, Manjunath [7] were able to feed a database of known skills to their CRF and as a result, it improved the model's performance. In our case, the model would have benefited from a database of common Spanish names to increase its performance in detecting people's names. The same could have been done if the model was trained for a more geographically restricted zone. For some entities, it is necessary to have the extra context to help the model learn better.

Regardless of the shortcomings, this model is a great basis for further work in the area. Since the model is good at discriminating essential resume information, it would already serve as a great tool to reduce the volume of data that recruiters and other HR representatives have to parse through. It could even be used as part of a parsing pipeline where its output is fed into another model to further refine the results. There is a lot of room for improvement but we can at least be sure that we are pointing in the correct direction.

There is a lot to gain from teaching computers to understand our language. The delegation of tedious and time-consuming work to automated models can free people to invest time in more complex and rewarding tasks. This project aimed to create a tool that could help people ease their workload, and to some extent, we were able to achieve that goal. We can only hope that, based on this research, others can build even more powerful tools and improve the collaboration between humans and automated systems.

REFERENCES

[1] B. Mohit, "Named Entity Recognition," in *Natural Language Processing of Semitic Languages*, pp. 221–245, Springer-Verlag Berlin Heidelberg, 2014.

[2] H. M. Wallach, "Conditional Random Fields: An Introduction," *University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-04-21*, nov 2004.

[3] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in *Proceedings of the 18th International Conference on Machine Learning 2001 (ICML 2001)*, pp. 282–289, 2001.

[4] S. Sanyal, N. Ghosh, S. Hazra, and S. Adhikary, "Resume Parser with Natural Language Processing," *International Journal of Engineering Science and Computing (IJESC)*, vol. 10, no. August, pp. 23–37, 2015.

[5] C. H. Ayishathahira, C. Sreejith, and C. Raseek, "Combination of Neural Networks and Conditional Random Fields for Efficient Resume Parsing," in *2018 International CET Conference on Control, Communication, and Computing (IC4)*, pp. 388–393, IEEE, 2018.

[6] P. K. Roy, S. S. Chowdhary, and R. Bhatia, "A Machine Learning approach for automation of Resume Recommendation system," *Procedia Computer Science*, vol. 167, no. 2019, pp. 2318–2327, 2020.

[7] S. H. E* and M. A E, "Differential Hiring using a Combination of NER and Word Embedding," *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 9, no. 1, pp. 1344–1349, 2020.

[8] J. Copara, J. Ochoa, C. Thorne, and G. Glavaš, "Spanish NER with Word Representations and Conditional Random Fields," in *Proceedings of NEWS 2016: 6th Named Entity Workshop at the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016*, pp. 34–40, 2016.

[9] M. Hilbert and K. Lu, "The online job market trace in Latin America and the Caribbean," tech. rep., Economic Commission for Latin America and the Caribbean (ECLAC), 2020.

[10] N. Okazaki, "Crfsuite: a fast implementation of conditional random fields (crfs)," 2007.

[11] H. B. Harvey and S. T. Sotardi, "The Pareto Principle," *Journal of the American College of Radiology*, vol. 15, no. 6, p. 931, 2018.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Since this is a classification problem, we can use different metrics to measure the performance of our model. We can obtain the precision, recall, and f-score by using a confusion matrix. Precision allows us to gauge how often our model correctly predicts," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011.

[13] D. M. W. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," pp. 37–63, 2020.