Multilingual Named Entity Recognition Solution for Optimizing Parcel Delivery in Online Commerce: Identifying Person and Organization Names

M. Pajas*, A. Radovan†, I. Ogrizek Biškupić‡
* BISS, Zagreb, Croatia
† BISS, Zagreb, Croatia
‡ Algebra University College, Zagreb, Croatia
matija.pajas@biss.hr, aleksander.radovan@biss.hr, ivana.ogrizekbiskupic@algebra.hr

Abstract—**This paper presents a comprehensive solution to enhance parcel delivery in online commerce by implementing multilingual named entity recognition. The solution is designed to accurately identify person and organization names, with a primary emphasis on correctly identifying recipients. The ultimate goal is to use this information to automatically validate recipients and select the most accurate one to improve data accuracy and reliability for parcel delivery. The process begins by collecting a large dataset of online commerce data, including customer search queries, and annotating it with person and organization names. The data is then preprocessed, cleaned to eliminate irrelevant information, and prepared for training a named entity recognition model. Next, the model is trained and evaluated using this data to ensure its ability to identify named entities and extract recipients from queries accurately. The process employs an iterative training process and data generation techniques, while also addressing the issue of noisy data and iterative training introducing unwanted patterns by retraining the model on the subset of the original annotated dataset. Our experiments conclude a consistent increase of $F1$ score over the baseline and best iteration using this method of training and fine-tuning.**

Keywords—named entity recognition, parcel delivery, data reliability, person and organization names, multilingual, natural language processing

I. INTRODUCTION

The increasing demand for online commerce has led to a need for dependable and efficient parcel delivery services. One way of reducing delivery expenses is to keep customer data as up-to-date and accurate as possible. However, there is a lot of noise in our database, especially in fields with recipient names that could be either personal or corporate, which comes from a vast number of accessible user inputs from various online shops, so the records and fields are not standardized. The noise appears in the form of misspelled words, irrelevant recipient information, and even multiple recipient names in a single record, making it challenging to match recipients with their correct addresses, leading to many duplicate records. To address it, this paper proposes a solution that leverages the latest advancements in natural language processing and machine learning to enhance parcel delivery in online commerce. The solution involves implementing a multilingual named entity recognition (NER) system that can accurately identify person and organization names, focusing on correctly identifying the intended recipient. The experiments and model selection for our named entity recognition task was performed using the FLAIR framework [1]. The FLAIR framework utilizes its proposed contextual string embeddings, known as *Flair Embeddings* [2] in the framework, which have proven to be highly effective for our downstream NER task. Additionally, self-supervised learning techniques [3], [4] and weak supervision were employed using Snorkel [5] to generate the training data as well as handwritten rules, and heuristics to generate additional synthetic data.

A. Recent research and challenges in real-world data

Named entity recognition involves identifying named entities, such as organizations, places, or individuals, from a large text corpus to extract knowledge and information automatically. Since early NER research papers were published using carefully handcrafted rules, such as Grishman and Sundheim [6] and Rau [7], the field has gained significant popularity. Collobert and Weston [8] introduced the first neural network architecture for NER, and since then, various neural network-based approaches have been proposed and published. The most significant development is the introduction of word and character embeddings used for word representation, improving sequence tagging tasks. This technique is now widely used in almost all NER studies [9]. Academic research in natural language processing (NLP) often concentrates on developing and improving recent neural network architectures, intending to slightly increase F1 scores on standardized datasets such as [10]. Although these efforts are crucial for advancing the scientific understanding of NER, they need to pay more attention to the practical application of these solutions on real-world data. Standardized datasets used for evaluation are often a good sign of a well-performing NER model and provide a common benchmark. Still, they need to reflect the complexity and diversity of real-world data that is often noisier and may not consider domain-specific knowledge that is critical for effective NER. This can result in NER models that perform well on standardized datasets but are unsuited for use in real-world data without adaptations.

The patterns that emerge in the data can vary greatly depending on the dataset being modeled. Models that performed great for some datasets, don't necessarily perform well on arbitrary datasets without necessary data processing or adaptation of model parameters. For instance, in the case of parcel delivery, if the data contains a large number of misspelled names or irrelevant information, it becomes difficult to extract the correct recipient name in automated processes, resulting in a higher number of failed deliveries and increased costs. On the other hand, if the data is well-structured and standardized, it becomes much easier to accurately identify and match recipients with their addresses, reducing the number of failed deliveries and lowering costs.

More often than not, NER tasks deal with unstructured and non-standardized text to extract useful information. Hence, it is essential to understand the structure and patterns of the data to build and train models effectively. Cheng et al. [11] proposed an end-to-end solution called *TripleLearn* to address a similar issue in their NER task of extracting brand and product types, which takes advantage of three different datasets: Golden training data (manually annotated data considered the best and unbiased source of labels), Noisy training data (generated from noisy customer data and product catalog), and Synthetic training data (generated using heuristics and rule-based generation). The authors also identify several related papers that use neural networks and a large amount of unlabeled data for commercial search engines [12]–[15]. They also demonstrate that the F1 score improves through iterative training. The exact details of how TripleLearn works and how it achieves improved performance are described in the paper. We extend the idea by incorporating the FLAIR, and Snorkel frameworks [1], [5] and tackle the problem of the iterative training process introducing new patterns that may not be relevant to the original dataset by retraining the best iteration on a subset of the original Golden Dataset, aiming to restore model's predictive ability on the original dataset. Using this method helps to restore knowledge about the initial problem while also retaining the insights gained from subsequent training iterations, thus leading to improved F1 scores and model accuracy. On GitHub [16], you can discover the demonstration version of the model as well as the outcome of the evaluation on a held-out test dataset.

II. Iterative training

A. Training dataset

Our final training dataset consists of various data sources, with the Recipient Delivery Database containing the primary dataset as depicted in Fig. 1. We focus on recipient names, including names in various languages, the most prominent being German, French, and Italian. The initial and most accurate Golden Dataset was created by sampling approximately 3000 data points from the database and having them annotated by domain experts. In addition to personal names, recipient names also include company names that are named in their respective language depending on the region. It is common to find person and company names in single data records, often combined with unnecessary information like phone numbers or addresses that were meant to be saved in other database fields. Moreover, the records often contain random junks or encoding errors that can adversely affect performance. Our goal is to extract entities of interest, person (PER) labels and organization (ORG) labels, in order to keep the fields containing recipient names as clean as possible. We aim to reduce this effect further through data preprocessing which leaves us with lowercased sentences containing only standard punctuation, letters, and numbers. That removes most of the noise and provides a standard data form for modeling.

After training the initial model using Golden Dataset, we use the model to data mine the underlying structure of a bigger database sample. We mainly consider the sequences in which entities are present in records to sample later datasets accordingly. Using this sequence distribution, we leverage Knowledge Bases containing the most common person and company names to generate Synthetic Dataset to mimic the rules and patterns of the main database to our best ability while also being utilized in the later process of generating new training datasets using the framework described below. By incorporating synthetic data we aim to cover a variety of different entities that are not necessarily contained in the dataset so the model can better learn the underlying entity patterns in the dataset.

B. Generating training dataset

The Snorkel framework [5] is a powerful tool designed to facilitate machine learning using weak supervision. Weak supervision refers to using multiple sources of noisy, imperfect data to train a model [17]. The Snorkel framework enables users to train machine learning models using weak supervision by generating training data from various sources of imperfect data. These sources can include different heuristic rules or comparisons with sources of truth, each of which will vary in their accuracy and coverage over the dataset being labeled. To accomplish this, the framework includes creating labeling functions (LFs), which label or abstain from labeling data points. These labels are then used to train a discriminative model using agreements and disagreements of LFs to probabilistically generalize training labels beyond what the LFs explicitly cover. In other words, the framework uses a combination of LFs to train a model that can generalize beyond the training set, despite the noise and imperfections in the data.

As part of this process, we also incorporate the model trained on the Golden Dataset as an (LF). By incorporating a NER model trained on Golden dataset into the Snorkel framework as an LF, training of the discriminative model can take advantage of the pre-existing knowledge and

TABLE I: Coverage and empirical accuracy of labeling functions (LFs) for sample size N=10000. Each LF has a certain polarity, labeling tokens as person (PER), organization (ORG) or both.

LF Function	Polarity	Coverage	Emp. Acc.
is company	ORG	0.1733	0.95
is company suffix	ORG	0.2376	0.96
is name prefix	PER	0.0364	0.7
is first name	PER	0.6976	0.78
is last name	PER	0.6148	0.73
pre-trained model	ORG and PER	0.9913	0.88

TABLE II: Training results showing training dataset size, accuracy, validation, and test $F1$ scores for every training iteration.

Iteration	Dataset size	Train Dev $F1$	Held-out Dev $F1$	Held-out Test $F1$
1	2944	0.993	0.928	0.916
2	27944	0.996	0.935	0.925
3	102944	0.975	**0.943**	**0.947**
4	202944	0.958	0.922	0.914
fine-tuned	2502	0.998	**0.964**	**0.953**

accuracy of the model. The idea is to secure correct labels with agreements between the current NER model and other LFs, but also challenge the outputs of the NER model in cases where its labels are inaccurate, but LFs disagree with it. This approach leverages the model's strengths while minimizing weaknesses, leading to a more robust and accurate discriminative label model. Finally, it's worth noting that to sample weakly labeled data, only high-confidence labels are considered, specifically those with a confidence score over 0.8. This ensures that the training data is of high quality and can be used to train a more accurate model. In the subsequent training iterations, a freshly trained model is employed as an LF, and the process is repeated.

We utilize a variety of other labeling functions (LFs) as sources of weak labels to generate training data in addition to the model itself. Table I lists several best LFs with their respective coverage and empirical accuracy, including LFs that identify common first names, last names, and companies while other LFs leverage expert knowledge of name and company prefixes and suffixes. For the specific scenario shown in the table, the model was trained solely on the Golden Dataset, meaning that the measurements provided in Table I pertain to the initial training iteration only.

C. Training

Our proposed training process is shown in Fig. 1. Inside the red rectangle is a repeatable training process where training data is generated through Snorkel framework before each training iteration. The process feeds each model with new weakly labeled data leveraging previous models knowledge and the statistical advantage of expert knowledge LFs ensemble combined to train the discriminative label model that has a final say in what the labels for each sentence should be.

The training starts by annotating a sampled Golden Dataset (1.) and starting the initial model training (2.) using it. After the model is trained, using Snorkel and synthetic data generation rules in parallel (3.) a combined training dataset is generated (4.) which is then combined with a Golden Dataset, starting a new training iteration (5.), closing the loop. Each iteration begins with a preprocessing and stratified sampling of all available datasets by class distribution and the most common patterns of entities. Particularly, we consider the order in which entities are placed inside each data record. We found the model is sensitive to the order of entities, and by sampling in this way, we avoid situations in which the model is classifying, for example the first entity as a PER instead of ORG based on the order bias. Furthermore, this process ensures that the sample drawn from each dataset is representative of the dataset's overall distribution as the order of words in a sentence can significantly alter its meaning. The resulting samples are then merged to form a single data frame labeled using Snorkel LFs and stratified sampled in the ways described.

The importance of data quality in training deep learning models must be considered. It has significant implications for the performance of the model. When it comes to weakly supervised learning, the initial dataset is particularly important, as it is used to develop the model and establish its rules. The concern regarding the introduction of noisy data patterns through the model's self-feeding of data persists. This could cause a deviation from the model's primary task of accurately predicting outcomes. Although we leverage data preprocessing and stratified sampling to minimize that effect, we observe that after a few training iterations, the models performance inevitably starts decaying due to poor quality of automatically generated labels. Another step we introduce to address this is using fixed contextual word embeddings as opposed to letting them fine-tune during the training as it is found to be preferable according to [18], because our model uses Flair Embeddings that are based on a language model similarly to ELMo that was used for embedding inputs for LSTM network in Arora et al. [18].

A similar observation in Cheng et al. [11] was made during their training experiments. The $F1$ score stopped improving after a certain number of iterations and started dropping significantly compared to previous iterations. In Table II, we can see that the $F1$ score drops below the initial score of a model trained on the Golden Dataset. The model started to learn self-created bias towards unknown patterns created during automatic data generation, which led to a decline in the models performance. This can be attributed to the fact that the model was trained on a combination of weakly supervised data, which unavoidably introduced noise and inconsistencies in the data that overwhelmed and completely declined the gain in accuracy after a certain point. To address this, we introduce one final step after iterative training before concluding the final score and deciding which model is going to be deployed.

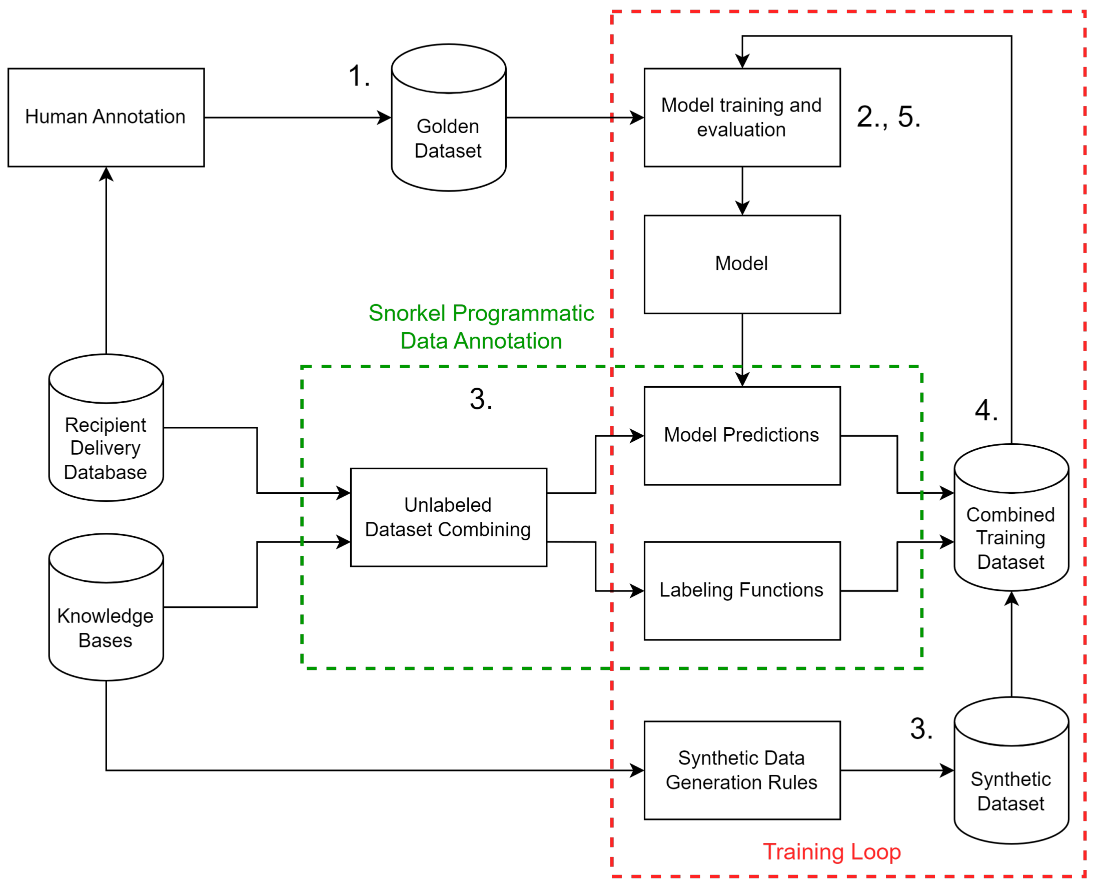
Human Annotation
1.
Golden Dataset
Model training and evaluation
2., 5.
Model
Snorkel Programmatic Data Annotation
3.
Recipient Delivery Database
Unlabeled Dataset Combining
Model Predictions
4.
Combined Training Dataset
Knowledge Bases
Labeling Functions
Synthetic Data Generation Rules
3.
Synthetic Dataset
Training Loop

Fig. 1: The process of creating a training set utilizing the Snorkel framework involves the stratified sampling of all datasets and merging them with labeled Golden Data to form a single data frame. This data frame is used with Snorkel to apply expert knowledge rules and compare tokens with knowledge bases while leveraging a pre-trained model developed using Golden Dataset. Once the model is trained on the Combined Training Dataset, it is utilized for Programmatic Annotation in the next training iteration.

III. Model fine-tuning

To fine-tune a model, first, we choose a model from training that is just one iteration before $F1$ score on our held-out validation set starts dropping. In our case, this is the third iteration as seen in Fig. 2. Finally, we proceed with fine-tuning the best iteration using 85% of the original Golden Dataset with a very small learning rate and up to 10 epochs in order to avoid overfitting and ultimately negatively affecting the model.

To select an appropriate learning rate, we choose a value from the optimal learning rate range where loss decreases the most and before it explodes as the learning rate becomes too large. This can be achieved by plotting loss as a function of learning rate and selecting the highest value of learning rate from the optimal range. This technique is described in Smith [19]. The technique can improve the performance of the model by finding an appropriate learning rate and avoiding problems such as slow convergence or overshooting the optima of the loss function. In Fig. 3, we can observe the plot of loss as a function of learning rate for our selected model. The steep optimal curve in the plot indicates that the model can still learn from the initial Golden Dataset. This finding further strengthens our suspicion that models trained in this way can in fact benefit from retraining on the Golden Dataset to address introduced bias through weakly labeled data generation. For our particular case shown in Fig. 3 we choose value 0.001 as our learning rate and fine-tune the model for 10 epochs. This results in a $F1 = 0.953$ on our held-out test dataset. The results are shown in Table II.

A. *Model Architecture and Word Embeddings*

Our approach employs a BiLSTM-CRF neural network which has demonstrated exceptional accuracy on NER datasets [20], [21]. We utilize contextual word embeddings, known as Flair Embeddings, accessible within the Flair framework [1], to generate character-based word embeddings that serve as inputs to the BiLSTM-CRF classifier, as shown in Fig. 4. The BiLSTM layer is a neural network that can process sequential data such as text, which works great for identifying sequences like named entities in sentences. This layer is designed to learn patterns and relationships between the input features, allowing it to capture complex patterns in the data. The CRF layer is a probabilistic graphical model used to

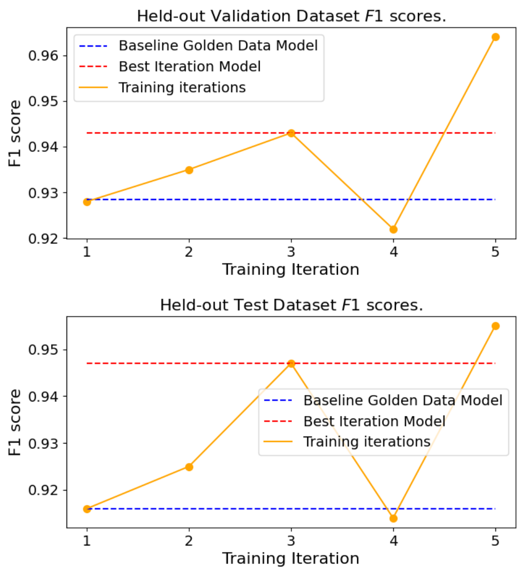

Fig. 2: Results from Table II include $F1$ scores for validation and test datasets (orange) per iteration, a baseline using only Golden Dataset (blue), and the best iteration's $F1$ score (red). The final measurement is from a fine-tuned best iteration model.

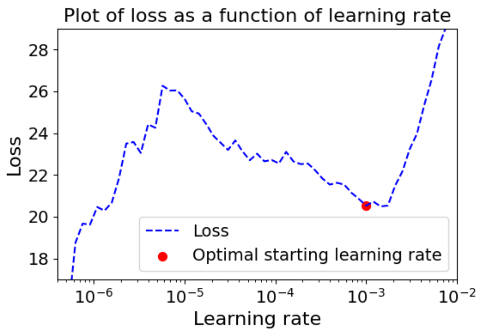

Fig. 3: A plot of loss against the learning rate using Adam optimizer. The red point shows the optimal starting learning rate for a fine-tuning process.

predict a sequence of labels based on the input features allowing the classifier to take into account the context of the words. The whole architecture is implemented through the Flair framework, which is built on top of PyTorch.

IV. Results

In Fig. 2, we observe that our final fine-tuned model, which is trained solely on the Golden Dataset, outperforms the baseline model and all other models in the iterative training process. To achieve this performance, we use our proposed solution, which involves generating additional data using our proposed techniques and fine-tuning the model from the best training iteration using a small learning rate and a subset of the initial Golden Dataset. This allows the model to relearn concepts from the initial training, while also improving the overall score and achieving higher $F1$ score than all the previous

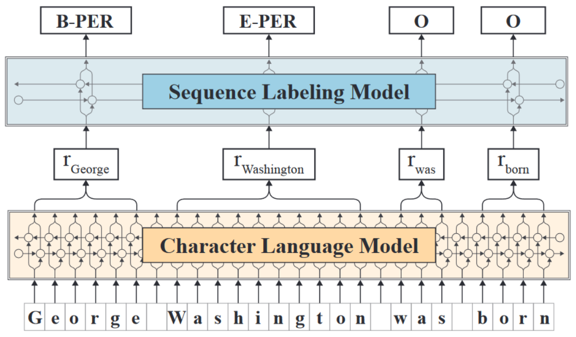

Fig. 4: The proposed approach involves using a pre-trained bidirectional character language model to obtain contextual embeddings of words in a sentence. These embeddings are then passed into BiLSTM-CRF sequence labeler [2].

training iterations. By fine-tuning the model on the original Golden Dataset, we can potentially restore its knowledge about the initial data while also retaining insights gained from subsequent training iterations and patterns imposed by noisy data generation.

The training results are summarized in Table II, where we observe that the iterative training process improved the $F1$ score by approximately 3.1% compared to the baseline. However, fine-tuning the best iteration on the original Golden Dataset resulted in an additional increase of approximately 0.6% in the $F1$ score. Consequently, the overall increase in $F1$ score from the baseline is 3.7%.

V. Future work

There are several avenues for future work related to our proposed approach. One promising direction is to explore knowledge transfer techniques to improve the initial performance of the model. This could involve leveraging knowledge from related tasks or domains to improve the accuracy of the model in the target task.

Another potential area of investigation is to explore the use of other advanced embedding models based on transformers and language models, or fine-tuning the existing ones. Fine-tuning these models could lead to even higher accuracy and better performance as the words for specific domain case would have better vector representations.

Finally, it would be interesting to investigate how our approach would affect other popular architectures such as BiGRU-CRF or CNN architectures. By experimenting with different model architectures, we can gain a better understanding of the generalizability and robustness of our proposed approach.

It would also be worth exploring the applicability of our approach to other domains. While our approach was developed and evaluated on a specific domain of identifying person and organization names of one particular multilingual country's delivery recipients, it has the potential to be adapted to other domains where the dataset is also multilingual, and similar challenges exist.

VI. Conclusion

Refitting a model on Golden Dataset in our proposed process of fine-tuning its parameters on the original dataset after iterative training, has a positive impact on its performance. It is important to keep in mind that the model would still retain some knowledge from training iterations, and this could potentially impact its performance on the original task. In our case, the retaining of knowledge from data generation techniques and training iterations shows an overall improvement of $F1$ score by 3.7% from the baseline, or by 0.6% from the best iteration, reaching $F1 = 0.953$. During our experiments this proved to be a consistent phenomenon.

One disadvantage of this method is its high computational cost due to training multiple iterations. However, our approach demonstrates that it is feasible to achieve state-of-the-art results in scenarios where a Golden Dataset annotated by human domain experts is a scarce resource. By leveraging a smaller dataset to establish initial knowledge and using an iterative training process that generates new data to train the model, we can overcome the challenge of the absence of human annotations. Furthermore, we can improve model accuracy by fine-tuning the model on the initial small Golden Dataset, which helps to restore the model's knowledge of the initial dataset. In addition to this, the knowledge retained from training iterations has proven to be beneficial as the overall accuracy of the model further improves after refitting. Although our approach comes with a high computation cost, we demonstrate that it is a viable alternative in situations where obtaining a large enough Golden Dataset for traditional supervised learning is not feasible.

Acknowledgment

We thank BISS d.o.o. for allowing us to use the hardware and equipment for training and testing purposes. Our gratitude goes to Dr. Matej Mihelčić, Matija's mentor, for providing guidance towards the final version of this paper.

References

[1] A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf, "FLAIR: An Easy-to-Use Framework for State-of-the-Art NLP," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 54–59. [Online]. Available: https://aclanthology.org/N19-4010

[2] A. Akbik, D. Blythe, and R. Vollgraf, "Contextual String Embeddings for Sequence Labeling," in *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 1638–1649. [Online]. Available: https://aclanthology.org/C18-1139

[3] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A Survey on Contrastive Self-Supervised Learning," *Technologies*, vol. 9, no. 1, p. 2, Mar. 2021, number: 1 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/2227-7080/9/1/2

[4] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using Self-Supervised Learning Can Improve Model Robustness and Uncertainty," Oct. 2019, arXiv:1906.12340 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1906.12340

[5] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: Rapid Training Data Creation with Weak Supervision," *Proceedings of the VLDB Endowment*, vol. 11, no. 3, pp. 269–282, Nov. 2017, arXiv:1711.10160 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1711.10160

[6] R. Grishman and B. Sundheim, "Message Understanding Conference- 6: A Brief History," in *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, 1996. [Online]. Available: https://aclanthology.org/C96-1079

[7] L. Rau, "Extracting company names from text," in *The Seventh IEEE Conference on Artificial Intelligence Application [1991] Proceedings*, vol. i, Feb. 1991, pp. 29–32.

[8] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," Jan. 2008, pp. 160–167.

[9] V. Yadav and S. Bethard, "A Survey on Recent Advances in Named Entity Recognition from Deep Learning models," Oct. 2019, arXiv:1910.11470 [cs]. [Online]. Available: http://arxiv.org/abs/1910.11470

[10] E. F. T. K. Sang and F. De Meulder, "Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition," Jun. 2003, arXiv:cs/0306050 version: 1. [Online]. Available: http://arxiv.org/abs/cs/0306050

[11] X. Cheng, M. Bowden, B. R. Bhange, P. Goyal, T. Packer, and F. Javed, "An End-to-End Solution for Named Entity Recognition in eCommerce Search," Dec. 2020, arXiv:2012.07553 [cs]. [Online]. Available: http://arxiv.org/abs/2012.07553

[12] D. Putthividhya and J. Hu, "Bootstrapped Named Entity Recognition for Product Attribute Extraction," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, Jul. 2011, pp. 1557–1567. [Online]. Available: https://aclanthology.org/D11-1144

[13] A. More, "Attribute Extraction from Product Titles in eCommerce," Aug. 2016, arXiv:1608.04670 [cs]. [Online]. Available: http://arxiv.org/abs/1608.04670

[14] M. Wen, D. K. Vasthimal, A. Lu, T. Wang, and A. Guo, "Building Large-Scale Deep Learning System for Entity Recognition in E-Commerce Search," in *Proceedings of the 6th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies*, ser. BDCAT '19. New York, NY, USA: Association for Computing Machinery, Dec. 2019, pp. 149–154. [Online]. Available: https://doi.org/10.1145/3365109.3368765

[15] A. Ahmed, C.-y. Wu, G. R. Kumar, and R. Datta, "Predicting Latent Structured Intents from Shopping Queries," 2017.

[16] "Github:mpajas/MIPRO-2023." [Online]. Available: https://github.com/mpajas/MIPRO-2023

[17] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National Science Review*, vol. 5, no. 1, pp. 44–53, Jan. 2018. [Online]. Available: https://doi.org/10.1093/nsr/nwx106

[18] G. Arora, A. Rahimi, and T. Baldwin, "Does an LSTM forget more than a CNN? An empirical study of catastrophic forgetting in NLP," in *Proceedings of the The 17th Annual Workshop of the Australasian Language Technology Association*. Sydney, Australia: Australasian Language Technology Association, 2019, pp. 77–86. [Online]. Available: https://aclanthology.org/U19-1011

[19] L. N. Smith, "Cyclical Learning Rates for Training Neural Networks," Apr. 2017, arXiv:1506.01186 [cs]. [Online]. Available: http://arxiv.org/abs/1506.01186

[20] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural Architectures for Named Entity Recognition," Apr. 2016, arXiv:1603.01360 [cs]. [Online]. Available: http://arxiv.org/abs/1603.01360

[21] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF Models for Sequence Tagging," Aug. 2015, arXiv:1508.01991 [cs]. [Online]. Available: http://arxiv.org/abs/1508.01991