Named Entity Recognition in Historic Legal Text: A Transformer and State Machine Ensemble Method

Fernando Trias*, **Hongming Wang, Sylvain Jaume, Stratos Idreos**

Harvard University

Abstract

Older legal texts are often scanned and digitized via Optical Character Recognition (OCR), which results in numerous errors. Although spelling and grammar checkers can correct much of the scanned text automatically, Named Entity Recognition (NER) is challenging, making correction of names difficult. To solve this, we developed an ensemble language model using a transformer neural network architecture combined with a finite state machine to extract names from English-language legal text. We use the US-based English language Harvard Caselaw Access Project for training and testing. Then, the extracted names are subjected to heuristic textual analysis to identify errors, make corrections, and quantify the extent of problems. With this system, we are able to extract most names, automatically correct numerous errors and identify potential mistakes that can later be reviewed for manual correction.

1 Introduction

Examining historical legal texts offers insight into the development of legal thinking and the practice of law. In order to facilitate computer processing, older legal texts are typically scanned from paper, microfilm or other physical media and then converted to text using Optical Character Recognition (OCR), which introduces numerous errors. Many of these errors can be corrected automatically using spelling and grammar correcting systems. However, the names of people, places and other proper names cannot be corrected easily, making the study of lawyers, judges and other people unreliable (Hamdi et al., 2020). One use of reliable names is inferring personal biases and connections that may affect outcomes (Clarke, 2018). In order to address this problem, names need to be accurately identified in the text and then corrected and standardized in a process often called Named Entity Disambiguation or NED (Yamada et al., 2016). Nonetheless, extracting accurate names is only part of the solution. In the future, organization names must also be extracted, and the respective roles must be identified.

The process of computationally extracting names from text is more formally called Named Entity Recognition (NER) and has been a difficult problem for many decades (Nadeau and Sekine, 2007). Furthermore, extracting names in legal text provides many domain-specific challenges (Bikel et al., 1999).

This paper describes a two-pronged approach for extracting the names of lawyers arguing cases:

(i) Extract the lawyer names from text using our ensemble model based on a neural network and a state machine.

(ii) Identify and correct transcription errors to uniquely identify lawyers.

Our system for extracting the names of lawyers in legal text uses a transformer-based neural network (Vaswani et al., 2017) feeding a finite state machine. After extraction, the identified names are subjected to several heuristic rules to identify errors, misspelled names and name variations in order to attempt to uniquely identify the lawyers named. When errors cannot be corrected automatically, such as in names with alternative spellings, the extent of the errors is quantified.

In order to develop, train and test this system, we used legal cases from the Harvard Caselaw Access Project (Harvard University, 2018). This project includes the complete text of decisions from United States courts dating back to the 1700s, with over 40 million pages of text spanning over 360 years. In our analysis, we only focused on cases from 1900 to 2010 in jurisdictions that were states as of 1900. Thus, Alaska and Hawaii were not considered. Because states and courts often have different

*Corresponding author. Email: fet535@g.harvard.edu, Harvard University, Cambridge, MA, USA

Natural Legal Language Processing Workshop 2021, pages 172–179

reporting styles that have varied substantially over the years, we segmented most of our analysis by state and then by decade.

2 Related Work

In a typical case text in the United States, lawyers are only identified in the header section on the first page using a relatively standardized format, usually called the "party names". They are rarely mentioned by name in the decision text. A typical party names text would read as follows:

> David P. Sutton, Asst. Corp. Counsel, with whom Charles T. Duncan, Corp. Counsel, Hubert B. Pair, Principal Asst. Corp. Counsel, and Richard W. Barton, Asst. Corp. Counsel, were on the brief, for appellant.

The text is usually a single complex sentence where all principal people, firms and their roles are identified in a mostly standardized and stylized format. Because of the sentence's complexity, the text is sometimes difficult for non-lawyers and automated systems to decipher. The parsing problem is compounded because the style standards and norms vary by location and over time. In addition to containing spelling and transcription errors, words and names are sometimes given as initials, nicknames or abbreviations. All these types of things confound automated systems.

Thus, a solution to the problem can be divided into two parts: (i) extract the names from the text and (ii) standardize the name to identify individuals.

One solution to the the first part of extracting names is documented by Dozier et al. (2010), who describe their work at Westlaw (now part of Thomson Reuters) in 2000 identifying entities in US case law using Bayesian modeling (Dozier and Haschart, 2000). Their process extracts more than just names and involves parsing words in part by using a finite state machine that is specially tailored for each jurisdiction. More recently, Wang et al. (2020) propose a solution based on a neural network architecture that performs well across various domains, including legal text. In addition, Leitner et al. (2019) have developed a very promising system to perform NER in German legal texts that was built and trained on their own dataset (Leitner et al., 2020). This dataset was also used by Bourgonje et al. (2020) in their NER work based on BERT. These approaches apply generically to the entire legal text and are not focused on the grammatically challenging party names text. In any case, lack of a similar dataset for English prevents us from trying these approaches.

Our system differs from previous attempts in that it is an ensemble composed of a transformer-based neural network and a state machine rather than a single architecture. The state machine allows the inclusion of pre-established knowledge of the syntax and style of the named parties text, thereby increasing accuracy. This increased the accuracy by 10% compared with the state-of-the-art transformer-based FLERT model (Schweter and Akbik, 2021).

3 Historical vs. Contemporary Names

Contemporary texts are usually digitally encoded at creation and thus do not suffer from OCR-related errors. In addition, many anachronisms such as the practice of using just initials have been supplanted over time so that currently almost all lawyers use their full name and middle initial. Abbreviations of names, such as "Geo." for "George", have also fallen out of favor. This simplifies the identification of names in contemporary texts.

In addition, contemporary names can be cross referenced with a standardized list of names such as Westlaw, Bloomberg, Martindale-Hubbell and similar directories that contain an almost comprehensive list of lawyers that can be used to uniquely identify individuals in the United States. However, such lists are subject to licensing restrictions and they have limited data for lawyers from the distant past.

Another consideration is that the same name can be used at different times and in different places to refer to different people. Thus, a name is generally only unique for a specific time and place. A related problem is when the same name is used by a parent and child and only differentiated by the use of "Sr.", "Jr.", "II", or similar suffix, if at all. Yet another related problem is the use of an initial instead of a first name. This problem will be discussed in more detail in section 5.2.

4 Model Architecture & Experiments

Our source code is available at `https://harvard-almit.github.io/legal-nlp`. To summarize, we developed a new system that combines a transformer neural network with

a finite state machine. In addition we trained an existing architecture with a subset of the Harvard Caselaw Access Project. These two were compared with the benchmark FLERT model.

- FLERT: Pre-trained general-purpose English-language NER model trained on CoNLL03.
- HCL-NER: Architecture based on FLERT but trained on Harvard Caselaw Access Project instead of CoNLL03.
- Ensemble: An ensemble model based on Flair's pre-trained PoS model and a custom state machine.

4.1 FLERT & Transformers

In recent years, numerous neural network models have been developed that can be used to perfrom NER, including Stanford NER, CMU, Flair, ELMo, BERT and many others. When we chose our benchmark to compare with our work, BERT was one candidate because it has been used extensively in a legal context, particularly for text classification by Chalkidis et al. (2020) in LEGAL-BERT. On the other hand, Flair (Akbik et al., 2018) is an easy-to-use Python framework that includes many pre-trained models, and provides more flexibility to extend our work at a later time. In addition to a BERT model, it includes it's own transformer-based model specifically trained for NER, which will serve as our benchmark, and another model for Parts of Speech (POS), which will be used by the ensemble model. Flair's NER model is called FLERT (Schweter and Akbik, 2021) and uses GLoVe (Pennington et al., 2014) global vectors for word embedding along with a Tranformer architecture based on XLM-RoBERTa (Conneau et al., 2020). Although FLERT performs well in ordinary text including the decision text, it does get tripped up by the party names text in the headers of legal decisions that name lawyers. To see why, consider the following example:

> Mr. Thomas A. McHarg, Messrs. Martin, Newcomer & Tinglof, Mr. Victor E. Keyes, attorney general, Mr. Bentley M. McMullin, assistant, Mr. Charles Roach, deputy, for defendant in error.

FLERT will return "Thomas A" and "Charles Roach". It misses McHarg's last name, "Victor E. Keyes" and "Bently M. McMullin". Thus, even though FLERT is tantalizing close, it is not sufficient if it cannot parse this relatively simple example.

Tag	Precision	Recall	F_1
LOC	0.9399	0.9434	0.9416
ORG	0.9014	0.8983	0.8998
PER	0.9600	0.9667	0.9634

Table 1: Validation results for HCL-NER model.

4.2 HCL-NER Model

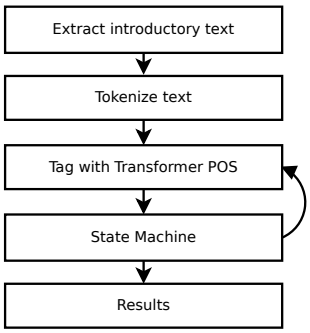
We hypothesized that one reason that FLERT does not perform well is that it is trained using the CoNLL03 dataset (Tjong Kim Sang and De Meulder, 2003), which consists of Reuters news stories from 1996 to 1997. Because the style of this training text is different from the style of text we aim to parse, we decided to train a new model with an identical architecture to FLERT, but using the Harvard Caselaw dataset instead of CoNLL03. We call this model HCL-NER. Because it duplicates the FLERT architecture, HCL-NER employs a multilingual XLM-RoBERTa (XLM-R) transformer-based model and GLoVe embeddings. The training, test and validation data comprised of 1000 cases selected randomly from the Harvard Caselaw Access Project. Of the 1000 cases, 100 were be reserved for development, 300 for training, 300 for testing and 300 for final validation of the model. These cases were parsed using an early version of the ensemble model described below and then manually reviewed and corrected. Three tags were used for tagging:

- LOC: A location, such as a city or state
- ORG: An organization, company or law firm
- PER: A person

Like FLERT, the model was trained using Stochastic Gradient Descent (SGD) for 150 epochs with the objective of maximizing the F_1 score (Sasaki, 2007). As will be shown later, this model performed slightly better than FLERT but did not perform as well as the ensemble model described below. We suspect that using a larger number of test cases in the future may improve performance.

The results of the HCL-NER model validation for each tag for are summarized in table 1.

Figure 1: Data pipeline for ensemble model



4.3 Ensemble Model

In addition to the HCL-NER model, we created an ensemble model that uses a pre-trained transformer to parse a sentence in order to identify Parts of Speech (PoS). That output is fed to a custom finite state machine that will represent knowledge of the writing style in order to identify the people, firms and roles of the individuals. Figure 1 shows a high level diagram of the ensemble. Although this paper focuses only on the names of people, the framework for identifying the firms and roles is also included, but not tested. This model performed the best of the various models tested.

The ensemble model consists of three distinct stages.

(i) Tokenize the text into distinct symbols and words.

(ii) Tag the tokens with the Part of Speech (PoS).

(iii) Pass the PoS and tokens to the state machine to extract items.

Tokenization is performed using Flair's default tokenizer, which is based on the 'segtok' library. PoS tagging is performed with Flair's standard PoS model, which is based on Long Short-Term Memory (LSTM) for sequence tagging along with a Conditional Random Field (CRF) layer (LSTM-CRF) (Huang et al., 2015). It identifies a number of different parts of speech. Of these, we are interested in:

- NN: Noun, singular or mass; in addition, Flair provides finer grain identification, such as NNP for a proper noun, NNPS for a plural proper noun, etc.
- ',' and '.': Punctuation symbols.
- IN: Preposition or subordinating conjunction.
- CC: Coordinating conjunction.
- Other: Other PoS elements are ignored for now, but they may become useful if the state machine is extended to identify the person's role and other data included in the text.

4.4 State Machine

In it's simplest form, the state machine transitions from one state to another based on the current state, the next token PoS, the token text (in the case of "Mr.'", "Hon.", etc.). However, there are a few cases where transitions will also look ahead several tokens to disambiguate particular esoteric cases. When the state transitions, it returns the text up to the transition marked with the state before the transition. The table 2 summarizes the transitions (including look-ahead exceptions in the notes). In the table, transitions are processed from top to bottom.

4.5 Abbreviations & Nicknames

Listed below are the three types of abbreviations that concern us. Each of these will be handled separately.

- Use of an initial instead of first name
- Abbreviations of words and titles (such as "Asst.", "Atty.", etc.)
- Abbreviations of names (such as "Geo.", "Thos.". etc.)

The simplest of these are the last two: the abbreviations of names, words and titles. These are identified by a period. When encountering a period, the previous word is then looked up in a table of known and typical abbreviations. If it is found, then the period is ignored since it does not mark the end of a sentence and the abbreviation is substituted with the corresponding word or name. Many databases of abbreviations exist for this purpose. In our example implementation, we used the list from Wiktionary (Wikitionary, 2021).

Nicknames are identified similarly to abbreviations. Once a name has been identified, the first name is looked up in a database of known basic nicknames and the formal name is substituted. In our implementation, the list from Northern and Nelson (2011) is used. However, care is required because in some cases, a nickname can be the actual name or can be used to differentiate two people with the same name.

Start State	Token	Condition	New State	Notes
MISC	NN	Match Mr. Hon. etc.	NAME	
MISC	NN	State name?	LOCATION	
MISC	NN	Abbreviation?	NAME	
MISC	NN	Modifier?	MISC	
MISC	NN	Starts with capital?	NAME	
MISC	DT	Starts with capital?	NAME	
NAME	NN	Abbreviation?	NAME	
NAME	‘,’	Look forward for Jr Sr?	NAME	
NAME	‘,’	Corp name?	COMPANY	
NAME	‘,’	City state?	LOCATION	
NAME	‘,’		MISC	End of name
NAME	CC	Is it ‘&’?	FIRM	
NAME	‘.’	Single or double initial?	NAME	
NAME	IN		MISC	End of name
FIRM	NN		FIRM	
FIRM	‘,’		FIRM	
FIRM	CC	Not ‘&’?	MISC	Firms use &
FIRM	Other		MISC	
COMPANY	NN		COMPANY	
COMPANY	Other		MISC	

Table 2: State transitions for ensemble model, one token at a time.

5 Performance Analysis

In this section, we evaluate the performance of our system regarding misspelled names and the use of initials instead of names and finally present a comparison analysis.

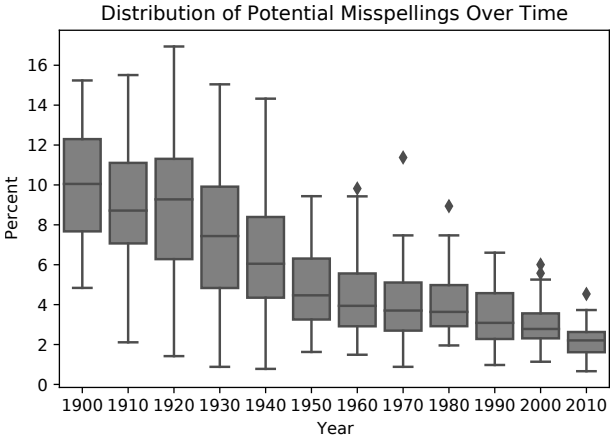
5.1 Misspelled Names

For all models, after a name has been extracted, it must be evaluated for possible misspellings due to OCR, transcription or other errors. These accidental misspellings are difficult to distinguish from deliberate alternate spellings. For example,“Clair”, “Claire”, and “Clare” are all possible misspellings of the same name, or they could be deliberate alternate spellings referring to different names. In the past, it was not uncommon for people to use various spellings for the same name depending on local conventions. For this reason, correcting for spelling mistakes in names is tricky and may produce more errors than it fixes if not done carefully. The problem of automatically adjusting misspelled names has been researched since at least the 1920s, beginning with Soundex, a system for encoding words phonetically. Famously, in the 1970s, the New York State Identification and Intelligence System (NYSIIS) (Silbert, 1970) attempted to solve this problem for criminal databases. Snae (2007) summarizes the results of many other name spelling matching systems, including NYSIIS. According to results from that paper, the Double Metaphone algorithm developed by Philips (Philips, 2000) for English seems to perform well in a variety of situations and especially with names.

The Double Metaphone algorithm works by creating a phonetic encoding of a word. To do so, it has a simplified alphabet of 16 consonants plus vowels. The algorithm is a sequence of steps that involve dropping and converting letters until the final encoding is achieved. For example, it begins by dropping duplicates and silent first letters using a few heuristic rules. Thus, for the first step, “written” becomes “riten”. A number of subsequent steps convert letter combinations with similar sounds to the same code. In this way, “sack” becomes “sak” and “enough” becomes “enouf”. Finally, all vowels except the first are removed. Thus “enough” is finally encoded as “enf”. In the example for “Clair”, “Claire” and “Clare” in the paragraph above, all encodings would resolve to “clr” and the algorithm would determine that they are all potentially the same name.

We used the Double Metaphone to match names in the database in order to count the number of potential misspellings and yield an estimate of the

Figure 2: Distribution of worst-case potentially misspelled names over time



worst-case scenario. For each state and decade, we created a list of all unique names in the dataset. Then we evaluated the Double Metaphone encoding for all names in each list and counted the number of differently spelled names that evaluate to the same encoding, using that number to calculate a percent of total names that are potentially duplicates. The boxplot in Figure 2 shows the distribution of this percent for each state and decade, which shows that the percent of potential misspellings is declining over time. This is expected due to the adoption of entirely digital creation and storage of data and the degradation of older original paper sources that result in more errors for older material.

5.2 Use of Initials Instead of Names

Another difficult problem is resolving the use of an initial as the first name. Although this was very common in some jurisdictions decades ago, the problem is almost non-existent today as shown by Table 3. The table compares the use of initials in 1900 to the use of initials in 2010. Massachusetts, the worst offender in 1900 at 79%, now has approximately only 0.1% use of initials. The highest level of current usage of initials is Kansas at 0.7%.

The problem with initials arises when they are ambiguous and could refer to one or more lawyers. Unfortunately, this problem is impossible to quantify without having a comprehensive list of lawyer names. However, a rough estimate of the extent of the problem can be calculated by looking at the names and seeing if the use of initials could refer to several lawyers whose full names are used elsewhere. In other words, we load up all the names and see if the names with only initials can match full names. Over time, different lawyers could in fact use the same initials as lawyers who practiced before, despite being different people. To mitigate this problem, the data was divided into states and then decades. Thus the comparison was made for every state and every decade. The results for the 10 worst state/decade combinations are shown in table 4.

Even though this is not the complete picture, it is possible to estimate that the problem of using an initial for the first name may not be too serious when identifying unique lawyers.

Table 3: Top 10 states by percent of lawyers using initials in 1900 vs. 2010

	% Names	
State	**1900**	**2010**
Massachusetts	79.0	0.1
Indiana	56.3	0.3
Maine	54.5	0.2
Georgia	48.1	0.4
Kansas	46.0	0.7
Arkansas	45.0	0.7
New Mexico	44.4	0.3
Vermont	44.0	0.3
Arizona	43.5	0.4
Wyoming	43.3	0.2

Table 4: Worst 10 states from 1900 to 2010 for using ambiguous initials shown as a percent of total initials used.

		Ambg.		%
State	**Year**	**init.**	**Total**	**Total**
Oregon	1940	2	1992	0.1
Indiana	1900	8	8763	0.1
Louisiana	1930	5	5612	0.1
Miss.	1940	2	2496	0.1
N. Carolina	1910	3	3882	0.1
Conn.	1900	1	1374	0.1
Oregon	1930	2	2797	0.1
Montana	1940	1	1399	0.1
Utah	1920	1	1542	0.1
Texas	1930	9	14979	0.1

5.3 Comparison

Model performance is compared using 10 random cases from each state, for a total of 490 cases. Each of these texts is reviewed manually to extract the names. First, in order to simplify evaluation and make it more explainable, the accuracy is measured as the percent of the test cases where the

Table 5: Simple accuracy comparison for the various model tested

Method	Errors	Total	Accuracy
FLERT	154	490	68%
HCL-NER	132	490	73%
Ensemble	107	490	78%

Table 6: Confusion matrices for various models tested

Method	TP	FP	FN	Prec.	Rec.
FLERT	622	81	159	0.88	0.80
HCL-NER	661	64	120	0.91	0.84
Ensemble	692	49	89	0.93	0.88

Columns are calculated as: *TP*, True positives; *FP*, False positives; *FN*, False negatives; *Prec.*, Precision: $\frac{TP}{TP+FP}$; *Rec.*, Recall or sensitivity: $\frac{TP}{TP+FN}$

model correctly identified all names. Next, recall and precision are calculated for all test cases. Table 5 summarizes the accuracy results. The ensemble model outperformed both FLERT trained on CoNLL03 and HCL-NER trained on a subset of the Harvard Caselaw Access Project.

Table 6 summarizes the confusion matrices for all test cases, with the following measures:

> *True positives (TP)*: Count of names in the original text correctly identified by the method.
>
> *False positives (FP)*: Count of identified names that were not valid names.
>
> *False negatives (FN)*: Count of names in the original text that were not identified by the method.
>
> *Recall (Rec.) or sensitivity*: True positives divided by all real positives (true positives plus false negatives). This is the portion of actual names that are correctly identified.

Both recall and precision for the ensemble models are improved relative to the FLERT benchmark. Thus, the ensemble model is able to more accurately identify all the names in the text and is also less likely to misidentify names. Although more true positives were identified, the biggest gains were in substantially fewer false negatives and positives. In addition, even though the HCL-NER model was not trained with a large dataset, it also is an improvement over the FLERT model.

6 Conclusion

We propose an ensemble model based on a transformer neural network architecture and a state machine for extracting names from party names in US case law text. The ensemble model improved the accuracy by approximately 10% from the FLERT model. However, once names were extracted, the problem of correcting for errors, misspellings and ambiguities could be fully automated. A number of techniques were discussed that help to quantify the extent of the problem and identify potential data quality issues. Our analysis showed that the number of errors in some jurisdictions could exceed 15% in the decades before 1950, but that this number has been declining significantly over time.

Acknowledgements

We would like to thank the Harvard Caselaw Access Project for providing us with support and access to their database.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th international conference on computational linguistics*, pages 1638–1649.

Daniel M Bikel, Richard Schwartz, and Ralph M Weischedel. 1999. An Algorithm that Learns What's in a Name. *Machine Learning*, 34:211–231.

Peter Bourgonje, Anna Breit, Maria Khvalchik, Victor Mireles, Julian Moreno Schneider, Artem Revenko, and Georg Rehm. 2020. Automatic induction of named entity classes from legal text corpora. In *ASLD 2020 – advances in semantics and linked data: Joint workshop proceedings from ISWC 2020. International workshop on artificial intelligence for legal documents (AI4LEGAL-2020), november 2-3, Athens/Virtual, greece*, pages 1–11. CEUR Workshop Proceedings.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. LEGAL-BERT: The Muppets straight out of Law School. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online. Association for Computational Linguistics.

Jessica A. Clarke. 2018. Explicit Bias. *Northwestern University Law Review*, 113:505.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco

Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Christopher Dozier and Robert Haschart. 2000. Automatic Extraction and Linking of Person. *Recherche dInformation Assistee parOrdinateur*, page 18.

Christopher Dozier, Ravikumar Kondadadi, Marc Light, Arun Vachher, Sriharsha Veeramachaneni, and Ramdev Wudali. 2010. Named entity recognition and resolution in legal text. In *Semantic processing of legal texts*, pages 27–43. Springer.

Ahmed Hamdi, Axel Jean-Caurant, Nicolas Sidère, Mickaël Coustaty, and Antoine Doucet. 2020. Assessing and Minimizing the Impact of OCR Quality on Named Entity Recognition. In *Digital Libraries for Open Knowledge*, Lecture Notes in Computer Science, pages 87–101, Cham. Springer International Publishing.

Harvard University. 2018. Caselaw Access Project. Retrieved 4 June 2020 from http://case.law.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *arXiv:1508.01991 [cs]*. ArXiv: 1508.01991.

Elena Leitner, Georg Rehm, and Julian Moreno-Schneider. 2019. Fine-Grained Named Entity Recognition in Legal Documents. In Maribel Acosta, Philippe Cudré-Mauroux, Maria Maleshkova, Tassilo Pellegrini, Harald Sack, and York Sure-Vetter, editors, *Semantic Systems. The Power of AI and Knowledge Graphs*, volume 11702, pages 272–287. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

Elena Leitner, Georg Rehm, and Julian Moreno-Schneider. 2020. A Dataset of German Legal Documents for Named Entity Recognition. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4478–4485, Marseille, France. European Language Resources Association.

David Nadeau and Satoshi Sekine. 2007. A Survey of Named Entity Recognition and Classification. *Lingvisticae Investigationes*, 30.

Carlton T Northern and Michael L Nelson. 2011. An unsupervised approach to discovering and disambiguating social media profiles. In *Proceedings of mining data semantics workshop*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Lawrence Philips. 2000. The double metaphone search algorithm. *C/C++ Users Journal*, 18(6):38–43.

Yutaka Sasaki. 2007. The truth of the F-measure. *Teach Tutor Mater*.

Stefan Schweter and Alan Akbik. 2021. FLERT: Document-Level Features for Named Entity Recognition. *arXiv:2011.06993 [cs]*. ArXiv: 2011.06993.

Jeffrey M. Silbert. 1970. The World's First Computerized Criminal-Justice Information-Sharing System - The New York State Identification and Intelligence System (NYSIIS). *Criminology*, 8:107.

Chakkrit Snae. 2007. A Comparison and Analysis of Name Matching Algorithms. *World Academy of Science, Engineering and Technology*, 25:6.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 142–147, USA. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*.

Jing Wang, Mayank Kulkarni, and Daniel Preotiuc-Pietro. 2020. Multi-Domain Named Entity Recognition with Genre-Aware and Agnostic Inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8476–8488, Online. Association for Computational Linguistics.

Wikitionary. 2021. Abbreviations for English given names. Retrieved on 30 June 2021 from https://en.wiktionary.org/wiki.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. *arXiv:1601.01343 [cs]*. ArXiv: 1601.01343.