Fall 11-9-2020

Ensemble labeling towards scientific information extraction (ELSIE)

Erin Murphy
DePaul University, EMURPH35@depaul.edu

ENSEMBLE LABELING TOWARDS SCIENTIFIC INFORMATION EXTRACTION (ELSIE)

BY

ERIN MURPHY

A THESIS SUBMITTED TO THE SCHOOL OF COMPUTING, COLLEGE OF COMPUTING AND DIGITAL MEDIA OF DEPAUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN DATA SCIENCE

DEPAUL UNIVERSITY
CHICAGO, ILLINOIS
2020

DePaul University
College of Computing and Digital Media

MS Thesis Verification

This thesis has been read and approved by the thesis committee below according to the requirements of the School of Computing graduate program and DePaul University.

Name: Erin Murphy

Title of dissertation: Ensemble Labeling towards Scientific Information Extraction (ELSIE)

Date of Dissertation Defense: 11/09/2020

Roselyne B. Tchoua

Advisor*

Daniela Raicu

1st Reader

Jacob Furst

2nd Reader

Alexander Rasin

3rd Reader

4th Reader (if applicable)

5th Reader (if applicable)

** A copy of this form has been signed, but may only be viewed after submission and approval of FERPA request letter.*

Abstract

Extracting scientific facts from unstructured text is difficult due to challenges specific to the ambiguity of the language, the complexity of the scientific named entities and relations to be extracted. This problem is well illustrated through the extraction of polymer names and their properties. Even in the cases where the property is a temperature, identifying the polymer name associated with the temperature may require expertise due to the use of acronyms, synonyms, complicated naming conventions and by the fact that new polymer names are being “introduced” to the vernacular as polymer science advances. While there exist domain-specific machine learning toolkits that address these challenges, perhaps the greatest challenge is the lack of—time-consuming, error-prone and costly—labeled data to train these machine learning models. Our work repurposes Snorkel, a data programming tool, in a novel approach as a way to identify sentences that contain the relation of interest in order to generate training data, and as a first step towards extracting the entities themselves. By achieving 94% recall and an F1 score of 0.92, compared to human experts who achieve 77% recall and an F1 score of 0.87, we show that our system captures sentences missed by both a state-of-the-art domain-aware natural language processing toolkit and human expert labelers. We also demonstrate the importance of identifying the complex sentences prior to extraction by comparing our application to the natural language processing toolkit.

1 Introduction

Extracting scientific facts from esoteric articles remains an important natural language processing (NLP) research topic due to the particularity of the entities and relations to be extracted. The challenges involved include the fact that entities can be described by multiple referents (synonymy) and conversely, the same word referring to different concepts depending on context (polysemy). Other nuances in the naming conventions render the extraction of these referents a non-trivial exercise even for curators as it requires specialized expertise. For example, in polymer science, there are distinctions between general and specific references to members of polymer families, or recognizing references to blends of two polymers, etc. Moreover, scientific relations are also complex as they may be one-to-many as opposed to single entity-to-entity relations, and require extra metadata to discern the exact relationship. Using the same polymer science example, a single scientific journal article may report on a polymer and several of its newly measured properties, and determining how these properties relate to the polymer can be complex. Similar issues arise in many fields as evidenced by NLP tools that rely on domain-specific grammar and ontologies.

Perhaps the most significant challenge in scientific NER is the lack of readily available labeled training data to enlist machines to the rescue. Indeed one of the major bottlenecks in developing machine learning-based information extraction is collecting large sets of hand-labeled training data. The process of creating well-balanced, manually-labeled datasets of scientific facts is more difficult due in part to the aforementioned challenges, but also due to the scarcity of entities and relations in scientific articles. For instance, it is not uncommon for scientists to write

an article about a newly discovered drug or newly synthesized material. To annotate sentences in such publications is not only tedious, error-prone and time-consuming, but also costly as it cannot easily be crowdsourced and requires time from subject matter experts.

Our ultimate goal is to alleviate the burden of expert annotators and facilitate complete and accurate extraction of scientific facts. Towards achieving this goal, we repurpose a data programming software [1] to identify sentences that contain scientific entities and relations automatically. Data programming is a paradigm for training models using higher-level, less precise supervision to avoid the bottleneck of collecting training data. Typically, this tool relies on existing entity taggers in order to identify and label relations. The key novelty of our approach is to identify sentences containing the target entities and relations without strictly identifying the entities through the use of dictionaries nor through complicated hard-coded rules. On a high-level, we identify sentences containing polymers and their associated glass transition temperatures without extracting the polymer entities. Instead, we use data programming to describe and combine approximate descriptions of the relations and the entities involved to recognize sentences of interest. We show that a computerized application built on combining weak, programmed rules, which target identifying specific entities can successfully identify sentences which contain such scientific entities, and scientific relationships regarding the targeted entities. Not only are sentences of interest overall accurately identified (94% recall), but the combination of weak, programmed rules are able to identify more sentences that are missed by human experts and state-of-the-art domain-specific computer software.

The rest of this paper is organized as follows. In Section 2, we describe background information on NER and material sciences which describe the unique challenges presented with polymer entity identification and extraction. Then we review research which attempts to solve problems related to ours, and highlight gaps in the current literature which we attempt to fill in our work. Section 3 presents the architecture of our application, which includes a discussion of the Snorkel Labeling Functions we developed, how they function when applied to the data, and a method of labeling data points of interest. Section 4 presents the results or our approach compared to state-of-the-art tools for scientific entity extraction and discusses significant findings and future work, followed by a conclusion in Section 5.

2 Background

Within NLP, there is a focus on identifying entities from unstructured data (text documents) that involves a number of tasks including: entity discovery, which detects names within text; named entity recognition (NER), which aims to identify the concept of an entity within text; relations extraction, which identifies the relation between named entities; and slot filling which identifies attributes of an entity [2]. It is hard to find tools that perform all tasks well and there is a lot of research done in each. There are two main kinds of extraction methods, one is based on knowledge engineering, and the other is based on machine learning. The former methods need domain experts to define and construct rules that utilize keyword matching, regular expressions, dictionaries, and ontologies. These rules are more or less complex and generally hard to maintain. The latter, currently more common are machine learning methods that require - often

manually - annotated corpora. While there are ways to leverage databases, semi-structured text from the Web or crowdsourcing for some applications, there is a critical need for especially designed alternatives for scientific information extraction. Some areas of medicine and biology benefit from an abundance of data through multiple databases[1]; other areas of bioinformatics and certain fields such as materials informatics still lack training data to fully leverage the advances in machine learning [3]. While this is a recognized challenge, there are few straightforward solutions to generating annotated scientific data as this process is tedious, error-prone and requires the costly focus and attention of experts in the field. Indeed, crowdsourcing is not viable in many cases due to the polysemy, synonymy and other esoteric nuances. In some cases, for example, authors are presenting new entities - not available in any dictionary - to their peers; in others they are describing in great details how entities and relationships are being measured (e.g. authors create new polymers and measure their properties).

2.1 Background on Material Sciences

In this section, we discuss the specificity of the polymer-glass transition extraction problem. Polymers are large molecules (macromolecules) composed of many repeating units, referred to as monomers. Due in part to their large molecular masses, polymers have a variety of useful properties [4]. Some of the useful properties of various engineering polymers are high strength or modulus to weight ratios (light weight but comparatively stiff and strong), toughness, resilience, resistance to corrosion and more. In fact, due to such properties, polymers are ubiquitous.

One specific property of polymers that has a profound impact on their application, is the glass transition temperature or Tg, which is the temperature at which a polymer transitions from a solid, amorphous, glassy state to a rubbery state as the temperature is increased. As the properties between the two states are drastically different, it is crucial to identify polymers with the appropriate Tg for different applications. For example, plexiglass (poly(methyl methacrylate)), used as a lightweight substitute for glass, has a high Tg of roughly 110 °C, while neoprene (polychloroprene), used for laptop sleeves, has a low Tg of roughly -50 °C [5]. Exact, as opposed to rough, values of Tg require additional contextual information such as the molecular mass. This is an example of metadata that could later be extracted after target sentences are identified.

2.2 Related Work

The medical community has long been invested in applying information extraction methods to medical publications [6-9]. These tools are designed to extract clinical information from text documents and to translate entities and terms to controlled ontologies and vocabularies. Other communities have followed, for example MedLEE [7,8] led to the development of more specialized tools such as GENIES [10] and BioMedLEE [11]. However, developing specialized ontologies, grammar and rules is error-prone, time consuming and hard to maintain. Moreover, it requires both a knowledge of the domain and in NLP.

[1] https://www.ncbi.nlm.nih.gov/

Recent scientific information extraction (IE) models remedy the above mentioned challenges by learning from data. While machine learning NLP techniques do not require the implementation of rich domain ontologies and grammars, they heavily rely on the availability of labeled training data. Statistical models such as Conditional Random Field (CRF) are graph-based models used in NLP to capture context by learning from sequences of words. Long short-term memory (LSTM) networks are recurrent neural networks that also capture context by learning relationships between a word and its preceding word. Bidirectional LSTM (Bi-LSTM) networks exploit information about the words that come before and after a given word. These models have shown great promise when applied to scientific IE [11-15]. For example, the ChemDataExtractor (CDE)--to which we compare our work and refer to as the state-of-the-art tool--implements an extensible end-to-end text-mining pipeline that can process common publication formats and produces machine-readable structured output data [12]. CDE automatically extracts chemical named entities and their associated properties, measurements, and relationships from scientific documents. It uses a combination of machine learning (linear-chain conditional random field) models, dictionary-based approaches, and regular expressions for entity recognition. Entity properties are extracted using a rule-based approach customized for specific properties. For such models to achieve high accuracy with regards to information and entity extraction, they require labeled data. This is especially limiting in material science. In fact, due to the lack of annotated data with high coverage of chemical data, the training data for CDE was supplemented using biomedical training corpora among other alternatives [12]. Finally, while tagging entities and identifying relations between them may be crowdsourced to the general public for general IE, labeling esoteric scientific articles requires domain knowledge, and therefore is more costly.

Distant supervision circumvents the need for expensive annotation by leveraging available databases or semi-structured text. For example deep learning tools such as PaleoDeepDive uses advanced statistical inference approaches to extract paleontological data from text, tables, and figures in scientific publications [16, 17]. For good performance, however, it first maps entities and their relations from a large database (PaleoDB at http://paleodb.org) to text. Unfortunately, many fields do not have access to such large databases of entities and relations, especially if new data is constantly being added to such databases.

Snorkel, for example, uses weak, programmed rules called labeling functions, to learn and model accuracies and conflicts between labeling functions to approximately create labels on unlabeled data, fast [1, 18]. Under certain conditions, applying data programming, such as using Snorkel labeling functions, achieves results on par with those of supervised learning methods. While Snorkel labeling functions can be used to approximately label large datasets, it is often used as a preprocessing step to label sentences that will be fed into an LSTM for example or assumes access to state-of-the-art entity taggers for relations extraction.
However, as previously mentioned, scientific entities and relations are complex and difficult to extract automatically; while many relations extraction work focuses on relations between two entities, scientific relations may consist of more entities and multiple relations or include

additional metadata [19,20]. Recent work introduced novel ideas of multiple relations extractions, however it refers to widely known general entities and relations [21].

In another example, computers cannot automatically extract entities that are not actually named in a publication (in which a new polymer is being synthesized). As a result, scientific IE often leverages a combination of crowdsourcing, machine learning, dictionaries and rules to extract relations from text [19,20]. In the T_g extraction pipeline, authors use a combination of automated methods to extract easily accessible T_g and crowdsourcing to extract more complex mentions as well as to review automatically extracted information [22]. Wallace et al. [23] use a hybrid machine learning and crowdsourcing approach to identify published randomized controlled trials (RCTs). They use machine learning classifiers to recognize citations that are deemed highly unlikely to describe RCTs, and defer to crowdsourcing otherwise [23]. In the GeneWays system, experts remove controversial collected data during the automatic literature extraction [24].

Our work uses Snorkel in a novel manner to address these crucial scientific IE challenges: 1) many NLP tools assume access to costly carefully labeled, balanced datasets, while in fact scientific entities are scarce in publications; 2) our entities are not always known a priori and are continuously being created or discovered; 3) relations identification is not dependent on first identifying the entities, and 4) our relations are complex and may contain more than one entity with multiple relations. In other words, all entities and relations might not be successfully extracted in an automatic fashion, but instead some relations will require further expert scrutiny in order to be extracted.

4 Architecture

4.1 Dataset

The input dataset contained 9,518 unique sentences from 31 journal articles containing “Tg” from a keyword search from the journal, *Macromolecules*, a prominent journal in polymer

Image 1. Example of Input database.[2]

[2] Extracted from: Mohanty, Angela D., Chang Y. Ryu, Yu Seung Kim, and Chulsung Bae. "Stable Elastomeric Anion Exchange Membranes Based on Quaternary Ammonium-Tethered Polystyrene-B-Poly (Ethylene-Co-Butylene)-B-Polystyrene Triblock Copolymers." *Macromolecules* 48, no. 19 (2015): 7085-95.

docid	text
acs.macromol.5b01382	Stable Elastomeric Anion Exchange Membranes Based on Quaternary Ammonium-Tethered Polystyrene-b-poly(ethylene-co-butylene)-b-polystyrene Triblock Copolymers
acs.macromol.5b01382	Angela D. Mohanty†, Chang Y. Ryu†, Yu Seung Kim‡, and Chulsung Bae†
acs.macromol.5b01382	Macromolecules, 2015, 48 (19), pp 7085–7095
acs.macromol.5b01382	DOI: 10.1021/acs.macromol.5b01382
acs.macromol.5b01382	Publication Date (Web): September 17, 2015
acs.macromol.5b01382	Copyright © 2015 American Chemical Society
acs.macromol.5b01382	*E-mail: baec@rpi.edu.
acs.macromol.5b01382	Abstract
acs.macromol.5b01382	A chemically stable and elastomeric triblock copolymer, polystyrene-b-poly(ethylene-co-butylene)-b-polystyrene (SEBS), was functionalized with various benzyl- and alkyl-substituted quaternary ammonium (QA) groups for anion exchange membrane (AEM) fuel cell applications.
acs.macromol.5b01382	Synthetic methods involving transition metal-catalyzed C–H borylation and Suzuki coupling were utilized to incorporate six different QA structures to the polystyrene units of SEBS.

science, during the years 2006-2016 [22]. The full text version of each article was downloaded in HTML format, and split into sentences (Image 1) so that each data point was a unique sentence tied to a document (journal article) identifier [22]. The journal article sentences were not preprocessed nor altered in any way such as removing special characters nor converting text from uppercase to lowercase.

3.2 A Priori Polymer Knowledge is Not an Option

As is often the case with entity extraction, external data sources such as databases or maps with known information about the entities of interest are used to help with identifying or tagging entities from text. As mentioned previously, a unique challenge with polymer data is that polymers are constantly being developed and their (often complicated) names are not known a priori to reading the text. Given this and the fact that databases containing information about polymers and their properties are not readily available, there exists a need to be able to extract polymers and their properties without relying on an external database to supply known information. In other words, a tool is needed which can not only extract polymer names from text without knowing them a priori, but also be able to extract information on the polymer's properties. We have, therefore, built, a tool with the knowledge (i.e. collection of simple labeling functions) to identify 1) polymers and/or their abbreviations, 2) a Tg mention, and 3) a temperature related to the respective polymer *and* Tg mention.

3.3 The Snorkel System and Its Built-In Functionalities

Snorkel is a system developed at Stanford University whose objective is to "...programmatically [build] and [manage] training datasets without manual labeling" [18]. In other words, it applies user-defined programmed rules as weak learners, to label data points in a dataset and avoids having to manually assign each data point. The weak learners, or rules programmed in a computer language such as Python, are known in Snorkel as Labeling Functions (LFs), which are considered one of the most important constructs of the Snorkel System. Multiple LFs can be created, and their logic can often be in opposition to each other, however, after applying LFs to the data, Snorkel can factor the patterns of how LFs interact with one another or the data point itself, and determine if a data point should be labeled or not.

The motivation behind using Snorkel for this project was to find a way to identify and extract polymer entities (polymer names/abbreviations and their properties, like Tg temperatures) in a fast and accurate manner compared to state-of-the-art tools and human experts in polymer science. Moreover, describing rules that identify and link entities related to one another is a more intuitive, adaptable and general way than writing sophisticated regular expressions to extract entity relations.

3.3.1 Snorkel Preprocessors and the Uniqueness of Polymer Data

The Snorkel Preprocessor is a particularly useful function which allows for each data point, or sentence, to be preprocessed in a user-defined manner. This is important because polymer names and entities do not always follow the same textual rules as do everyday English texts. For example, abbreviations of polymer names, which are largely uppercase alpha character strings, are ubiquitous throughout polymer texts. Applying a preprocessing function to make all text lowercase before applying the Snorkel LFs would therefore result in not being able to identify an abbreviation. However, there are times when the same sentence containing an abbreviation needs to be made lowercase in order to find a different entity in the sentence, such as the mention of a glass transition temperature. For example, “glass transition” can be denoted as “TG” or “Tg” or “tg” within and/or across texts as there is no accepted rule across the community as to how “glass transition” is abbreviated. Take the following sentence into consideration:

Bacterial polyhydroxy alkanoates such as poly(3-hydroxybutyrate) (P3HB), poly(3-hydroxyvalerate) (P3HV), or higher hydroxy acids and their copolymers display decreasing melting points from about 180 °C (Tg = 1−4 °C) for P3HB to 112 °C (Tg = −12 °C) for P3HV. [3]

As mentioned, to find a *glass transition* mention by searching for “tg” (in order to catch any transformation of “TG” or “Tg” or “tg”), the sentence could simply be made lowercase, and a character string search for “tg” could be performed. But if the act of making all text lowercase was permanent, then finding *polymer abbreviations* within that same preprocessed sentence would be impossible if the rules to find a *polymer abbreviation* state that they are **uppercase**, mostly-alpha character strings, such as “P3HB” and “P3HV.”

Finding the different entities, *polymer abbreviations* and *glass transition mentions* therefore require completely different impermanent, preprocessing rules on the same data points, which are easily accommodated by the Snorkel Preprocessor function. When a LF is executed, it can call one or many Snorkel Preprocessors. For example, when LFs looking for a *glass transition* are executed, the Snorkel Preprocessor to make text lowercase is called only for that LF, and after processing, the sentence resumes its original, unaltered state. This allows a second LF looking for a *polymer abbreviation* to be executed on the unaltered sentence as found in the original publication.

[3] Sentence extracted from: Petrovic, Zoran S, Jelena Milic, Yijin Xu, and Ivana Cvetkovic. "A Chemical Route to High Molecular Weight Vegetable Oil-Based Polyhydroxyalkanoate." *Macromolecules* 43, no. 9 (2010): 4120-25.

Three preprocessors are built for this work: *makeTextLower()*, *makeCharUniform()* and *removeSpacesInParentheses()*. The *makeTextLower()* is self-explanatory in that it converts the data point being processed into lowercase. The *makeCharUniform()* was created because special characters such as dashes and apostrophes can appear and be formatted as different characters when read in from various journal articles. For example, a dash can be represented by the following characters:

1) -
2) ‒
3) –
4) —

There are definitive differences between the actual lengths of 1) and 4), whereas the differences between 2) and 3) are more difficult to discern. When placing the characters side-by-side and making them bold the appear as follows:

- ‒ – —

This comparison illustrates that the pixels of 2) and 3) sit at different heights in a row and therefore are really two different characters. The reason this is important is because one of the entities of interest in this work is temperature values which can sometimes be <0°, and some of the LFs account for negative signs. If a negative sign appears as the character “-” in one paper and “—” in another, it is more difficult to write rules which look for every variation of a negative sign. Instead, a Snorkel Preprocessor was written to take all variations of a dash and convert it to a uniform dash.

The final preprocessor, *removeSpacesInParentheses()*, is also unique for polymer text because polymer names can often contain multiple character tokens within parentheses such as the polymer name: poly(tetrafluoroethylene). Although this is the common spelling for this polymer, it is possible that a spelling or format mistake occurs in which the polymer is referred to as: poly(tetrafluoro ethylene). If this were the case, it would be important that a computer program knows that both poly(tetrafluoroethylene) and poly(tetrafluoro ethylene) are the same polymer. Therefore, instead of removing all spaces from a sentence, a preprocessor was built to remove spaces only within parentheses to account for this specific example.

3.3.2 Labeling Functions (TRUE, JUNK, ABSTAIN)

When Snorkel LFs are applied to data points, they return values of 1, 0 and -1 indicating that a LF returns a TRUE label, a FALSE (JUNK) label or ABSTAINS, respectively. There is a certain degree of attention that must be paid to this step given that it can greatly impact the overall rating of labeling a sentence as TRUE or FALSE. For example, if three LFs are assigned to label as sentence, and the output of those three LFs on the one data point render the results of [1, 0, 1], it is clear that 2 of the 3 LFs have deemed the sentence to be TRUE (1), if using a majority voting system. When the LF’s output are [1, 0, 0], the sentence would be deemed FALSE (0) since the majority of outputs deemed the sentence to not be TRUE. However, if the

LF's output are [1, 0, -1], this is equivalent to saying that only two LFs produced a label and one is not counted, or abstains. There is then a 50% chance the data point is TRUE or FALSE. Therefore, this latter scenario could either be prone to missing TRUE data points or assigning FALSE data points.

Later, we will review how this was considered when extracting the entities of interest. This also points out that care must be taken not only when generating LFs, but also the values that are returned from the LFs after processing a data point. Snorkel provides a number of tools which assist the user in researching LF outcomes, and should be utilized when generating and applying LFs.

3.3.3 LFs to Identify Different Entities

To be labeled as TRUE, a sentence must contain 3 different and unique entities: a *polymer name* or its *abbreviation*, a *glass transition* mention, and the *glass transition temperature of the respective polymer*. Snorkel uses LFs to process and label an entire datapoint which results in outputting a label of TRUE, FALSE or ABSTAIN for the respective data point. This work leverages this functionality, but instead of all LFs being used to determine if the data point should be labeled as TRUE or not, we group the LFs into buckets so that each bucket represents the presence of (or lack thereof) one of the three entities. As a result, each LF looks for only one entity, and groups of LFs collectively aim to identify only one of the three entities based. If a group of LFs identify the presence of its respective entity, and each of the three entities are found in a sentence, only then is the sentence labeled as TRUE. If only one or two entities are present, the sentence is not labeled as TRUE.

An *ensemble labeler* was developed in response to determining whether all entities are present. It reads in the Snorkel LF output arrays, decides if all three entities are present, and if so, labels the sentence as TRUE. This ensemble labeler will be discussed later on in this section. The following three sections describe the LFs which search for each of the three desirable entities.

3.3.4 Labeling Polymer Entities

Finding the presence of a polymer and/or its abbreviation without a priori knowledge of its name sounds like a daunting task, however, we are able to identify sentences with polymer entities with only four LFs. In other words, we show that entities can be found without the use of external reference dictionaries feeding knowledge to the LFs or writing rules which use extensive REGEX functions. It should also be emphasized that even though the below four LFs indicate that a polymer name is identified in a sentence, the sentence will only be returned as TRUE if a *glass transition mention* and a *temperature* are also identified in the same sentence. Below is the listing of the LFs which identify the existence of a polymer entity and descriptions of their logic.

Table 1. List of Polymer-Identifying LFs

LF Name	Description
abbreviation_in_sentence()	This LF looks for a token within a sentence that consists only of uppercase alpha characters, numbers and special characters. Only 40% or less of the token can consist of special and numeric characters. For example, **P3HB** is considered an abbreviation, whereas **270°C** is not since 100% of characters in the latter token are numbers and special characters. If the criteria is met, the LF returns 1, otherwise it returns a -1. Note that it would not be appropriate to return a 0 if the logic is not met because there are some polymers that do not have abbreviations, and we do not want to penalize the sentence for not containing an abbreviation; we therefore simply abstain.
keyword_poly()	This LF looks for the character string, “poly” in a sentence. If it exists, a 1 is returned, otherwise a -1 is returned.
keyword_polyParen()	Similar to keyword_poly(), if a sentence contains, “poly(“, then a 1 is returned, otherwise a -1 is returned.
keyword_copolymer()	There are naming conventions applied to certain types of polymers known as copolymers. This LF accounts for those rules in that if any of these character strings are found in a sentence, a 1 is returned, else a -1 is returned. Examples of character strings found in copolymers are: "-co-", "-stat-", "-per-", "-ran-", "-grafted-", "-trans-", and "-alt-".

Essentially, the LFs above are looking for abbreviations, keywords of “poly” or “poly(“ and character strings indicating a copolymer. We have found that these four LFs are sufficient at identifying polymer names and/or abbreviations.

3.3.5 Labeling Tg Temperature Entities

It is simple to identify numbers in a sentence, but it is more difficult to discern what those numbers represent. The challenge in finding sentences that contain temperature entities has a silver lining in that temperatures are most often followed by degree (°) symbols. This idea is relied upon in a number of our LFs to identify temperature entities, but there are other rules that need to account for numbers followed by a degree (°) symbol that do not represent a temperature, for example the size of an angle.

Again, for a sentence to be returned as TRUE, the LFs need to have indicated that all three entities of interest are identified in a sentence. A simple way to immediately negate a sentence is to check if any numbers exist in a sentence; if no numbers are found, then a temperature will not be present in a sentence. If a number is found, this rule should only return a -1 (ABSTAIN) and not a 1 because returning a 1 would indicate that a suspected temperature is present. Numbers in a sentence indicate a temperature presence is possible, and the sentence should not be considered JUNK; it is the role of other LFs to determine if the numbers in question actually represent a temperature.

Table 2. List of Temperature-Identifying LFs	
LF Name	**Description**

tempUnits()	This LF simply looks for a degree (°) symbol. If found, it returns 1, otherwise it returns -1.
tempUnitsAfterNumber()	If a degree (°) symbol is not present after a temperature, a unit of temperature must be present, such as C (Celsius), F (Farenheit) or K (Kelvin) to indicate the numbers represent a temperature. If the numbers are immediately followed by a C, F or K, then a 1 is returned, otherwise a -1 is returned.
tempUnitsAfterDegree()	A combination of the prior two LFs, if a degree (°) symbol is followed by a C, F, or K, then this is a strong predictor that a temperature exists in the sentence and a 1 is returned, otherwise a -1 is returned.
equalSignBeforeNumber()	If an equal (=) sign exists before numbers (with or without special characters like - or ~, as are sometimes associated with temperatures), then a 1 is returned, otherwise a -1 is returned.
circaSignBeforeNumberDegree()	Similar to the above LF, if the tokens “circa” or “ca” or “about” precede a number (with or without special characters like - or ~, as are sometimes associated with temperatures), then a 1 is returned, otherwise a -1 is returned.
tempRange()	Glass transition temperatures associated with a polymer can be reported as a temperature range. Therefore, this LF returns a 1 if more than 40% of a token’s characters consists of numbers, such as in the case of “-2 - -1” which could read, “negative 2 to negative 1.” Otherwise a -1 is returned.
JUNKtempUnitsAfterNumber()	If a number exists and is not followed by a degree (°) symbol, C, F, or K, then the number is assumed to not be a temperature and a 0 is returned, otherwise a -1 is returned.
JUNKtempUnitsAfterDegree()	If a degree (°) symbol exists in a sentence and is not followed by a C, F, or K, then it is assumed the sentence does not contain a temperature and a 0 is returned, otherwise a -1 is returned.
JUNKnoNumbers()	If there are no numbers in a sentence, then a 0 is returned, otherwise a -1 is returned. A 1 is not returned because a 1 represents an assumption that a temperature exists. Since not all numbers represent temperatures, it can only be assumed that a sentence containing numbers is at, best, not a JUNK sentence.

3.3.6 Labeling Tg Mentions

There are a discrete number of ways that a *glass transition mention* can be expressed through text, which is either by spelling out “glass transition” (with varying forms of capitalization), shortening it to “glass trans” or “glass-trans”, or abbreviating it to simply “tg.” Ultimately, this search can be streamlined to searching for: “glass t” or “glass-t” or “tg.”

However, in polymer texts there is a technique called thermogravimetric analysis, which is sometimes abbreviated as, “TGA.” Therefore, additional LFs needed to be created to distinguish sentences that contain “TGA” vs just “TG” to avoid labeling sentences that only refer to TGA as containing a *glass transition mention* entity.

Table 3. List of Glass Transition Mention-Identifying LFs	
LF Name	**Description**
keyword_tg()	If the character strings “glass t” or “glass-t” or “tg” are found in a sentence a 1 is returned, otherwise a -1 is returned.
JUNK_tga()	If the character string “TGA” is found in a sentence a 0 is returned, otherwise a -1 is returned.
JUNK_tgAndTGA()	This is considered a “tie-breaker” LF for sentences containing “TGA.” If this LF didn’t exist, then sentences with “TGA” would return output arrays as [1, 0] and would need to be resolved with a tie-breaker (i.e. randomly assigning the *glass transition mention* entity as 1 or 0). Therefore, if “TG” is found in a sentence with no other alpha characters following it, a 1 is returned; if the character string “TGA” is found, then a 0 is returned; otherwise a -1 is returned.

3.4 Majority Ensemble Labeler and ELSIE

As mentioned above, the labeling functions of the Snorkel system ultimately consider all outputs of the LFs for a respective data point to determine if that data point will be labeled as TRUE or not, noting that the all LFs are considered in combination to label the respective data point. The difference between our work is that before determining if a data point (sentence in this work) is labeled TRUE, the output values of the LF first determine if its respective entity is present in the sentence, and if all entities are present, then the sentence is labeled as TRUE.

In total, there are 16 LFs used in this work where an output value of 1 = TRUE, 0 = JUNK, and -1 = ABSTAIN. The first four LFs, highlighted below in yellow, aim to identify *polymer names and abbreviations*, the next nine LFs, highlighted in green, aim to identify *temperatures* and the last three, in blue, aim to identify *glass transition mentions*. Visually this can be represented as values in the output array in conjunction with the following sentences where the color-coding of the LFs below corresponds to the entities identified in the sentence.

LFs to Identify Polymer Names and Abbreviations
abbreviation_in_sentence, keyword_poly, keyword_polyParen, keyword_copolymer,

LFs to Identify Temperatures
TempUnits, tempUnitsAfterNumber, tempUnitsAfterDegree, equalSignBeforeNumber, circaSignBeforeNumberDegree, tempRange, JUNKtempUnitsAfterNumber, JUNKtempUnitsAfterDegree, JUNKnoNumbers,

LFs to Identify Glass Transition Mentions
keyword_tg, JUNK_tga, JUNK_tgAndTGA

Sentence 1
Bacterial polyhydroxy alkanoates such as poly(3-hydroxybutyrate) (P3HB), poly(3-hydroxyvalerate) (P3HV), or higher hydroxy acids and their copolymers display decreasing melting points from about 180 °C (Tg = 1−4 °C) for P3HB to

112 °C (Tg = −12 °C) for P3HV.[4]

Output labeling matrix: [1, 1, 1, -1, 1, 1, 1, 1, -1, -1, -1, -1, -1, 1, -1, 1] → [3/3, 4/4, 2/2]

For Sentence 1, of the LFs that did not abstain (where the output was either a 1 or 0, but not a -1), all three entities of interest were identified in the sentence; 3 of 3 LFs that did not abstain found a *polymer name or abbreviation* (yellow), 4 of 4 LFs that did not abstain found a *temperature* (green), and 2 of 2 LFs that did not abstain found a *glass transition mention* (blue).

Sentence 2

Although the corresponding copolymers were afforded with perfectly alternating nature and excellent regiochemistry control, only glass-transition temperatures of around 8.5 °C were observed in the differential scanning calorimetry (DSC) curve, demonstrating that the polymers are completely amorphous (see Supporting Information Figure S3).[5]

Output labeling matrix: [1, 1, -1, -1, 1, 1, 1, -1, -1, -1, -1, -1, -1, 1, -1, -1] → [2/2, 3/3, 1/1]

Similar to sentence 1, LFs applied to Sentence 2 also indicate that all three entities of interest are found in the sentence. To note, more LFs ended up abstaining with this sentence than in Sentence 1, but the end result is that both sentences contained all three entities.

Sentence 3 (shown 3 times to illustrate how Snorkel is able to correctly label tricky sentences)

1) The TGA scans indicated that APNSi has 5% decomposition in air of 340 °C and in argon of 450 °C (Figure 1).
2) The TGA scans indicated that APNSi has 5% decomposition in air of 340 °C and in argon of 450 °C (Figure 1).
3) The TGA scans indicated that APNSi has 5% decomposition in air of 340 °C and in argon of 450 °C (Figure 1).[6]

Output labeling matrix: [1, -1, -1, -1, 1, -1, 1, -1, 1, -1, 0, -1, -1, 1, 0, 0] → [1/1, 3/4, 1/3]

In Sentence 3, Snorkel LFs identified an abbreviation of “TGA.” Because there are no sources for distance learning, Snorkel LFs cannot determine if the abbreviation represents a polymer or not, so the output LFs are only able to demonstrate that an abbreviation is identified. A temperature entity is obviously identified with the appearance of numbers followed by “°C”. Finally, the LFs found the string, “TG” in the sentence and will need to discern if that represents a glass transition mention or not. The LFs are able to determine that a *Tg mention* is not present in the sentence (refer to 3.3.6 Labeling Tg Mentions for further clarification). Ultimately, only a *polymer name or abbreviation* and *temperature* entities were confirmed to be found in Sentence

---

[4] Extracted from: Petrovic, Zoran S, Jelena Milic, Yijin Xu, and Ivana Cvetkovic. "A Chemical Route to High Molecular Weight Vegetable Oil-Based Polyhydroxyalkanoate." *Macromolecules* 43, no. 9 (2010): 4120-25.

[5] Sentence extracted from: Yue, Tian-Jun, Wei-Min Ren, Ye Liu, Zhao-Qian Wan, and Xiao-Bing Lu. "Crystalline Polythiocarbonate from Stereoregular Copolymerization of Carbonyl Sulfide and Epichlorohydrin." *Macromolecules* 49, no. 8 (2016): 2971-76.

[6] Sentence extracted from: Finkelshtein, E Sh, KL Makovetskii, ML Gringolts, Yu V Rogan, TG Golenko, LE Starannikova, Yu P Yampolskii, VP Shantarovich, and T Suzuki. "Addition-Type Polynorbornenes with Si (Ch3) 3 Side Groups: Synthesis, Gas Permeability, and Free Volume." *Macromolecules* 39, no. 20 (2006): 7022-29.

3, and the LFs could not confirm that a *glass transition mention* was identified. Therefore, the sentence was not labeled as TRUE. This final example demonstrates the power of Snorkel Labeling Functions and how the combination of weak learners allow the system to carve out the entities of interest while ignoring entities not of interest from the sentence by picking up on nuances of rules to discern which sentences to label, even tricky ones.

Once the output arrays of the LFs are generated and it is determined if each of the entities are present or not, the ensemble labeler determines which sentences should be labeled as TRUE or not. The ensemble labeler uses a simple majority of LF outputs per entity, and combines these results to determine if a sentence should be labeled as TRUE or not. Again, if and only if all entities are present in a sentence will the ensemble labeler label the sentence as TRUE. As a result, we are calling this process of considering the output of all LFs per entity then determining if all entities are present, *ensemble labeling toward scientific information extraction*, or ELSIE.

4 Results and Analysis

We first discuss how the initial gold standard dataset--labeled by human experts, and the dataset from the state-of-the-art tool--which normally aims to extract data, were generated so they could be compared to sentences labeled by ELSIE. Next we discuss how the initial gold standard dataset is updated after being compared to the ELSIE's output which revealed TRUE sentences that were missed by human experts. Finally, the state-of-the-art tool's and ELSIE's outputs are both compared to the updated gold standard (hereafter referred to as the "gold standard") dataset. It should be noted that the state-of-the-art tool's performance in identifying sentences of interest against the gold standard is discussed as a matter of comparison to ELSIE's performance and labeling abilities.

4.1 Training Dataset and its Labels

To test how well ELSIE identified sentences of interest--those containing a *polymer name or abbreviation*, a *temperature* and a *glass transition mention*, we compared the output results to the same document corpora that was labeled by human experts (the initial gold standard) [21]. The state-of-the-art tool's ability to label sentences of interest was also compared to the ELSIE's performance in labeling sentences.

An important note about our initial gold standard dataset and the state-of-the-art tool's dataset is that the intention of both was to extract polymer entities and their glass transition temperatures, which differs from the current intention of ELSIE's which aims to label sentences containing three entities: a *polymer name or abbreviation*, a *temperature* and a *glass transition mention*. The motivation behind our approach is that scientific entities and relations in target sentences can be too complex to exclusively be automatically extracted and may require additional human attention as illustrated in this section. To align the initial gold standard and the state-of-the-art tool-labeled data with ELSIE-labeled data, metadata about sentences and polymer-Tg pairs extracted by experts and CDE was used to automatically label sentences in the documents they were extracted from. Data correctly extracted by the state-of-the-art tool was previously

validated by experts as well [21]. If the state-of-the-art tool extracted a polymer-Tg pair correctly, the sentence(s) from which the information was obtained by the state-of-the-art tool were labeled as 1; if the state-of-the-art tool extracted an incorrect polymer-Tg pair (i.e. an incorrect polymer was paired with a Tg temperature), sentences containing the correct polymer name or abbreviation and the Tg mention were both labeled as 0 [21]. Sentences identified by the human experts which contained polymer-Tg pairs were labeled as 1. If a polymer-Tg mention existed in the corpora, and the human experts and/or the state-of-the-art tool did not extract the pair, sentences were labeled as 0.

4.2 Updated Gold Standard Labels

After running ELSIE on our unlabeled dataset, we discovered that there were sentences not labeled as TRUE in the initial gold standard dataset that should have been labeled as TRUE (i.e. they contained a *polymer name or abbreviation*, a *temperature* and a *glass transition mention* but were not labeled as TRUE). We considered these to be false “false positives” from the initial gold standard dataset in that they were TRUE sentences that were missed by human experts. Further details of these sentences is provided later in section 4.3, but as a result of these findings, the dataset was updated to reflect the corrected label of TRUE to these previously “missed” sentences, and it is this updated dataset--the gold standard--to which the state-of-the-art tool and ELSIE are compared.

4.3 Results

The final document corpora contained 9,518 sentences (data points), representing 31 unique scientific journal articles. Overall, the state-of-the-art tool labeled 15 sentences as positive cases, ELSIE labeled 67 sentences as positive cases, and the human expert identified 49 sentences as positive cases. The gold standard dataset contained 64 positive cases.

Positive cases represent less than 1% of the data, illustrating the highly unbalanced nature of the dataset, and therefore the accuracy of comparing the performance of the state-of-the-art tool and ELSIE to the gold standard does not convey the entire story of each application's performance. Precision and recall results, along with accuracy and F1-scores, are presented in Table 4.

Table 4. Performance Compared to the Gold Standard

	Gold Standard	Human Experts	State-of-the-Art Tool	ELSIE
Total Cases	9,518			
Total Positive Cases	64	49	15	67
Accuracy		99.84%	99.49%	99.88%
Precision		100%	100%	90%

Recall		77%	23%	94%
F1 score		0.87	0.38	0.92

The analyses were run on a personal laptop computer using Python 3.6 in Jupyter Notebook. The total processing time to process all 9,518 sentences through ELSIE, including Snorkel preprocessors, was 0:01:03, compared to the state-of-the-art tool's processing time which took 0:26:00 to process 31 documents.

Table 5. Confusion Matrices

Gold Standard versus State-of-the-Art Tool					**Gold Standard versus ELSIE**				
		Actual					**Actual**		
		Positive	**Negative**				**Positive**	**Negative**	
Predict	**Positive**	15	0	15	**Predict**	**Positive**	60	7	67
	Negative	49	9454	9503		**Negative**	4	9447	9451
		64	9454				64	9454	

4.4 Analysis

Comparing the F1 scores of the state-of-the-art's performance to the gold standard (0.38) versus ELSIE's performance to the gold standard (0.92) shows that ELSIE is better at labeling sentences correctly in terms of identify all three entities of interest: a *polymer name or abbreviation*, a *temperature* and a *glass transition mention*. There are a number of reasons for ELSIE's superior performance to the state-of-the-art tool which requires a closer look at both methods' precision and recall.

Precision for the state-of-the-art tool was higher than ELSIE's because ELSIE labeled sentences as TRUE when they should not have been labeled as such (i.e. false positives). This was because ELSIE was looking for entities within a sentence (even if the entities were not all related to one another), whereas the state-of-the-art tool was looking specifically for related entities. Overall, the number of sentences labeled by the state-of-the-art tool was much smaller (n=15) than ELSIE (n=67), with no sentences considered as false positives with the state-of-the-art tool, whereas 7 of the 67 labeled sentences by ELSIE were false positives. An example of a false positive sentence labeled by ELSIE is shown in Exhibit 1, where *polymer name* and *glass transition mention* entities were identified in the sentence, but the temperature entity identified in the sentence was regarding the polymer's melting temperature and not the glass transition temperature. Though it is a false positive, reporting this sentence may be beneficial in some

regards because it could contain important metadata either about the three polymer entities of interest, or other characteristics of the polymer.

Exhibit 1. False Positive Sentence (Labeled TRUE by ELSIE)

Sentence	Gold Standard	Human Experts	State-of-the-Art Tool	ELSIE
Two or three thermal transitions are expected for SEBS: (1) a low glass transition temperature (Tg1) corresponding to the ethylene-co-butylene block, (2) a high glass transition temperature (Tg2) corresponding to the styrene block, and (3) a broad endothermic transition at the melting temperature (Tm) near 20 °C, depending on the degree of crystallinity of the ethylene-co-butylene block. [7]	0	0	0	1

As mentioned above, ELSIE correctly identified TRUE cases that the human experts and the state-of-the-art tool had missed (see Exhibit 2). As a result, the gold standard dataset was updated to reflect the TRUE cases, and recall for the human experts (77%) was lower than that of ELSIE (94%). This illustrates that a high level of attention is required by humans (even human experts regarding this subject matter) when reading texts otherwise there is a chance that important information can get missed. This finding also highlights ELSIE's robustness in and reliability in labeling scientific [polymer] sentences for training data over human experts and state-of-the-art tools aiming to perform the same function.

Exhibit 2. Sentences Missed by Human Experts, Labeled by ELSIE

Sentence	Gold Standard	Human Experts	State-of-the-Art Tool	ELSIE
Upon 10 wt % clay loading, the glass transition of the PTMO:MDI−BDO PU nanocomposites shifts slightly from −44.7 to −46.6 °C.[8]	1	0	0	1

[7] Extracted from: Mohanty, Angela D., Chang Y. Ryu, Yu Seung Kim, and Chulsung Bae. "Stable Elastomeric Anion Exchange Membranes Based on Quaternary Ammonium-Tethered Polystyrene-B-Poly (Ethylene-Co-Butylene)-B-Polystyrene Triblock Copolymers." *Macromolecules* 48, no. 19 (2015): 7085-95.

[8] Extracted from: James Korley, LaShanda T, Shawna M Liff, Nitin Kumar, Gareth H McKinley, and Paula T Hammond. "Preferential Association of Segment Blocks in Polyurethane Nanocomposites." *Macromolecules* 39, no. 20 (2006): 7030-36.

The functionalized polycarbonate exhibited a lower Tg of 89 °C compared to its parent (108 °C).[9]	1	0	0	1

Because it is more important for this work to capture all true labels and return false positives than it is to miss true labels, it is acceptable for precision to be compromised for the sake of obtaining high recall. ELSIE's ability to label sentences missed by state-of-the-art tools and human experts demonstrates the power weak learners play in capturing [polymer] entities in text. They are able to identify nuances in the data because they are a collection of weak learners; nuances are the exception to the rule, and collections of weak learners cater to nuances.

The state-of-the-art tool's recall (23%) is much lower than ELSIE's recall (94%) because the state-of-the-art tool missed labeling more positive cases (n=49) than ELSIE (n=4). Again, given that the state-of-the-art tool's objective was to extract entities and not label sentences, when the state-of-the-art tool extracted an incorrect polymer-Tg pair, it was penalized and the sentence was not labeled. It is important to note that the state-of-the-art tool would have achieved higher recall (88%) had we focused only on rule-based extraction of Tg. However due to the nuances in complex sentences and complicated polymer naming, it often linked the Tg to incorrect polymer names [21].

In this work, ELSIE missed labeling sentences where all three entities of interest were not contained within a single sentence. This demonstrates how and why the problem of finding polymers and their respective glass transition temperatures is hard for computers and easier for humans. Exhibit 3 shows an example where the first sentence only contains a *polymer* entity (which should be noted that ELSIE was also able to identify), yet it did not contain a *temperature* nor *glass transition* entities; the human identified this sentence and received credit. To note, the state-of-the-art tool extracted the Tg mention, but paired it to the wrong polymer, and therefore did not receive credit. The other two entities are found in the next sentence where the human experts also received credit (again, human experts mapped the polymer name to the correct Tg-mention). Since all three entities were spread among multiple sentences and not contained within one sentence, ELSIE was not able to label the sentences as TRUE.

Exhibit 3. True Positive Sentences Missed by LFs

Sentence	Gold Standard	Human Experts	State-of-the-Art Tool	ELSIE
The azo-polymer material, poly[4'-[[2-(acryloyloxy)ethyl]ethylamino]-4-nitroazobenzene], often	1	1	0	0

[9] Extracted from: Darensbourg, Donald J, Wan-Chun Chung, Andrew D Yeung, and Mireya Luna. "Dramatic Behavioral Differences of the Copolymerization Reactions of 1, 4-Cyclohexadiene and 1, 3-Cyclohexadiene Oxides with Carbon Dioxide." *Macromolecules* 48, no. 6 (2015): 1679-87.

referred to as poly(disperse red 1 acrylate) (hereafter pdr1a), was synthesized as previously reported.[10]				
The prepared material was determined to have a molecular weight of 3700 g/mol, and a corresponding Tg in the range 95−97 °C.[11]	1	1	0	0

4.5 Discussion and Future Work

An application using Snorkel system functionalities, like its preprocessors and labeling functions, was developed to process polymer-related scientific journal articles and identify sentences containing polymer entities and their glass transition temperatures. This application, ELSIE, specifically looks for three separate and unique entities: polymers, temperatures and glass transition mentions. Though this work focuses specifically on polymer-related text and entities, the foundations of this work and the concepts of identifying multiple entities (some which may have never been seen or tagged in texts) with ELSIE can be used in domains reaching far beyond Material Science.

Work concerning polymer sciences is lacking training data in terms of volume and completeness compared to other fields where efforts to collect training data have been occurring longer and have had greater contributions by experts in the field. Polymer science illustrates an additional scientific challenge in that new polymers (or words) are continuously being created (or introduced into the vocabulary), and therefore it is almost impossible to completely rely on an external source or database to reference all polymers that exist or approximately label data. As polymers are created, the information is most often contained in published journal articles which can be time-consuming for humans to read and a challenge for computers to keep track of and process. As a result, databases containing polymer information are often out-of-date, expensive to maintain, and incomplete. Though state-of-the-art computer technologies used to extract polymer-related information from text exist, these systems often rely on coding extensive rules to extract polymer information from text and have been shown to be inaccurate at times. Therefore, a more reliable and efficient computer-assisted application that is able to identify polymer information from texts without needing a priori knowledge has been created to fill this gap.

This work focuses on identifying entities which are not known a priori to text processing instead of looking for a relationship-type word which relates two or more known entities together (i.e. oxidized, married, friend). A sentence will only receive a label of TRUE if it contains all three entities of interest. The purpose for this restriction is to imply a relationship (in that the distance of the words are proximal given that each entity is found in the same sentence) without having

[10] Extracted from: Yager, Kevin G, and Christopher J Barrett. "Photomechanical Surface Patterning in Azo-Polymer Materials." *Macromolecules* 39, no. 26 (2006): 9320-26.

[11] Extracted from: Yager, Kevin G, and Christopher J Barrett. "Photomechanical Surface Patterning in Azo-Polymer Materials." *Macromolecules* 39, no. 26 (2006): 9320-26.

to look for a word or words defining a relationship between the three entities. To determine a glass transition temperature of a polymer, the polymer, a temperature and something implying the temperature is a glass transition all need to be present within a single sentence. To concretely relate all three entities together is more complicated since there are many ways to imply a relation among the three entities, and beyond the scope of this paper. Future work will need to iterate over and process the data multiple times to 1) identify and isolate sentences of interest and their surrounding sentences, and 2) extract polymer entities and their properties from the isolated sentence population. This work has already begun, and we are currently able to show that polymer entity extraction from these isolated sentences is possible. This is supported by the fact that Snorkel LFs used in this work are grouped together to target and identify mutually exclusive entities. The identification of these entities by Snorkel LFs therefore begets entity extraction.

With regards to 1) of future work, not only are the sentences of interest isolated, but consideration should be taken to process sentences prior to and preceding the sentence of interest. It is known that sentences of interest do not always contain all three entities, but instead one sentence could contain a polymer entity and the following sentence(s) might contain the polymer's glass transition temperature. As seen earlier, ELSIE struggled with identifying these sentences because the majority ensemble labeler restricted the criteria which stated a sentence will only get labeled TRUE if it contains all three entities. Therefore, when working on polymer entity extraction, additional sentences, aside from those containing the three entities of interest, will need to be considered.

Along with entity extraction, isolating all sentences of interest plus their surrounding sentences allows for more robust information extraction such as pointing out information of interest so that a human does not have to read the entire article or highlighting sentences that might otherwise be missed within the text. It is also possible that these sentences contain valuable metadata the human needs. For instance, extracted as opposed to approximate Tg values require additional contextual information such as the molecular mass. Also, understanding the rules for entity identification can potentially assist or guide authors in writing text in a way that is easier or more obvious for a computer to parse out the entities of interest.

This work has also shown that state-of-the-art systems and human experts are still susceptible to missing sentences containing polymer information. Using Snorkel LFs in ELSIE allows for nuances and multiple types of complexities to exist in a sentence and is still able to accurately identify sentences of interest. One reason nuanced and complex sentences are missed by humans and computers is because they can contain multiple polymers, or the polymer and/or their properties are referenced as something other than their name or abbreviation (i.e. Tg1, Tg2). Also, human experts are still human, and there is always a real possibility that they can miss information. If a computer application can reliably process polymer texts and not suffer from fatigue or attention-loss, and do so at faster speeds, then it is well worth the effort and investment in these systems.

5 Conclusion

An application using Snorkel system functionalities, including its preprocessors and labeling functions, was developed to process polymer-related scientific journal articles and identify sentences containing polymer entities and their glass transition temperatures. The Snorkel LFs and majority ensemble labeler to create ELSIE used in this work represent a collection of simple and easy to understand programmed rules that are able to handle nuances in the data to distinguish entities of interest between tokens not representing an entity of interest.

The application's Snorkel LFs are able to identify entities without a priori knowledge of the entities, which is particularly useful when dealing with polymers given that external data sources are often incomplete and out-of-date. Additional functionality was built to only label sentences which met the criteria of containing all three entities of interest.

Our application had a recall of 94% when compared to the gold standard, though struggled to find sentences where three presumably-related entities existed in multiple sentences. However, our application found sentences missed by computers and human experts whether due to sentences being complicated, such as a sentence that contains multiple polymer-Tg pairs, or fatigue/lack of attention paid by human experts. Ultimately, we show that this work is able to build robust labeled training datasets, begets the ability to perform entity extraction for polymer data, and can ultimately be used to further study extracted entity relationships.

References

[1] Ratner, Alexander J., et al. "Data programming: Creating large training sets, quickly." Advances in neural information processing systems. 2016.

[2] U.S. National Institute of Standards and Technology (NIST). "TAC Knowledge Base Population (KBP) 2017," February 14, 2018. Available at: https://tac.nist.gov/2017/KBP/index.html.

[3] Deyu Zhou and Yulan He. Extracting interactions between proteins from the literature. Journal of Biomedical Informatics, 41(2):393–407, 2008.

[4] Brinson, Hal F., and L. Catherine Brinson. "Polymer engineering science and viscoelasticity." An introduction (2008).

[5] J. Brandrup, E. H. Immergut et al., Eds., Polymer Handbook, 4th ed. Wiley-Interscience, 1999.

[6] Jagannathan, V., and A. S. Elmaghraby. "MEDKAT: multiple expert DELPHI-based Knowledge Acquisition Tool." Proceedings of the ACM NE Regional Conference. 1985.

[7] C. Friedman, P. O. Alderson et al., “A general natural-language text processor for clinical radiology,” Journal of the American Medical Informatics Association, vol. 1, no. 2, pp. 161–174, 1994.

[8] C. Friedman, G. Hripcsak et al., “Representing information in patient reports using natural language processing and the extensible markup language,” Journal of the American Medical Informatics Association, vol. 6, no. 1, pp. 76–87, 1999.

[9] G. K. Savova, J. J. Masanz et al., “Mayo clinical text analysis and knowledge extraction system (cTAKES): Architecture, component evaluation and applications,” Journal of the American Medical Informatics Association, vol. 17, no. 5, pp. 507–513, 2010.

[10] C. Friedman, P. Kra et al., “GENIES: A natural-language processing system for the extraction of molecular pathways from journal articles,” in ISMB (supplement of bioinformatics), 2001, pp. 74–82.

[11] L. Chen and C. Friedman, “Extracting phenotypic information from the literature via natural language processing,” Studies in Health Technology and Informatics, vol. 107, no. 2, pp. 758–762, 2004.

[12] Swain, Matthew C., and Jacqueline M. Cole. "ChemDataExtractor: a toolkit for automated extraction of chemical information from the scientific literature." Journal of chemical information and modeling 56.10 (2016): 1894-1904.

[13] Rocktäschel, Tim, Michael Weidlich, and Ulf Leser. "ChemSpot: a hybrid system for chemical named entity recognition." Bioinformatics 28.12 (2012): 1633-1640.

[14] D. M. Jessop, S. E. Adams et al., “OSCAR4: A flexible architecture for chemical text-mining,” Journal of Cheminformatics, vol. 3, no. 1, p. 41, 2011.

[15] Hong, Zhi, et al. "SciNER: Extracting Named Entities from Scientific Literature." International Conference on Computational Science. Springer, Cham, 2020.

[16] De Sa, Christopher, et al. "Deepdive: Declarative knowledge base construction." *ACM SIGMOD Record* 45.1 (2016): 60-67.

[17] S. E. Peters, C. Zhang et al., “A machine reading system for assembling synthetic paleontological databases,” PLoS One, vol. 9, no. 12, p. e113523, 2014.

[18] Ratner, Alexander, et al. "Snorkel: Rapid training data creation with weak supervision." Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases. Vol. 11. No. 3. NIH Public Access, 2017.

[19] Tchoua, Roselyne B., et al. "Blending education and polymer science: Semiautomated creation of a thermodynamic property database." Journal of chemical education 93.9 (2016): 1561-1568.

[20] Tchoua, Roselyne B., et al. "A hybrid human-computer approach to the extraction of scientific facts from the literature." Procedia computer science 80 (2016): 386-397.

[21] Liu, Jin, et al. "Multiple relations extraction among multiple entities in unstructured text." Soft Computing 22.13 (2018): 4295-4305.

[22] Tchoua, Roselyne B., et al. "Towards a hybrid human-computer scientific information extraction pipeline." 2017 IEEE 13th International Conference on e-Science (e-Science). IEEE, 2017.

[23] B. C. Wallace, A. Noel-Storr et al., “Identifying reports of randomized controlled trials (RCTs) via a hybrid machine learning and crowdsourcing approach,” Journal of the American Medical Informatics Association, 2017.

[24] Rzhetsky, Andrey, et al. "GeneWays: a system for extracting, analyzing, visualizing, and integrating molecular pathway data." Journal of biomedical informatics 37.1 (2004): 43-53.