Weakly Supervised Extraction of Tasks from Text

Sachin Pawar, Girish K. Palshikar, Anindita Sinha Banerjee
TCS Research, Pune, India.
{sachin7.p,gk.palshikar,anindita.sinha2}@tcs.com

Abstract

In this paper, we propose a novel problem of automatic extraction of tasks from text. A task is a well-defined knowledge-based volitional action. We describe various characteristics of tasks as well as compare and contrast them with events. We propose two techniques for task extraction – i) using linguistic patterns and ii) using a BERT-based weakly supervised neural model. We evaluate our techniques with other competent baselines on 4 datasets from different domains. Overall, the BERT-based weakly supervised neural model generalizes better across multiple domains as compared to the purely linguistic patterns based approach.

1 Introduction

We define a *task* as a well-defined knowledge-based action with a specific goal and which is carried out within a small time period, often by a single person, a group of persons, a device, or a system. The tasks usually demand some skill and expertise by the human actor(s) performing the task and they are carried out volitionally by the actor(s).

The problem of *Task extraction* is to automatically identify mentions of such tasks in text. Syntactically, a task can be mentioned as a verb phrase (e.g., `implemented a model for weather prediction`) or as a noun phrase (`model implementation for weather prediction`) in a sentence. Table 1 shows various examples of tasks observed across multiple domains (also see Table 6 for a comprehensive list of tasks). The extent of a task mention should be such that the complete meaning expressed by the task should be captured. For example, from the sentence `The researcher implemented a model for weather prediction.`, it is expected to identify the entire phrase `implemented a model for weather prediction` as a task, even though the shorter phrase `implemented a model` is a valid task mention but does not capture the entire meaning.

Event extraction (Xiang and Wang, 2019) is a popular task in NLP literature. An *event* is generally defined as a specific occurrence of something happening in a certain time and place which involves one or more participants and can often be described as a change of state. Although events are similar to tasks in some aspects, there are certain crucial distinctions (described in detail in Section 2) and hence it is important to define and address task extraction as a separate problem.

There are several interesting analyses that can be carried out over the extracted tasks from large corpora. Similar tasks can be mentioned in different ways and it is important to cluster the tasks together which have similar *meanings*. By definition, a task needs certain expertise to be carried out and it would be an interesting problem to determine difficulty level for each task. Usually, each task is carried out by an actor and this actor often plays a certain generic *role* such as engineer, banker, farmer etc. Also, most tasks need certain skills to be carried out such as various technical or domain concepts, programming languages, certain tools or technologies, etc. Such co-occurrence or interdependence between tasks and roles as well as between tasks and various skills, can be studied.

Task extraction has several useful real-life applications. For example, tasks extracted from *resumes* capture the fine-grained experience of the candidate and would be quite useful for automatically shortlisting candidates for a certain job requirement. Another interesting application of the extracted tasks and their corresponding roles is to automatically augment common sense knowledge. For example, ConceptNet (Speer et al., 2017) contains knowledge of the form ⟨ `Engineer`; *is capable of*; building a bridge⟩ or ⟨ `Policeman`; *is capable of*; arresting criminals⟩. But the number of such triplets are lim-

Proceedings of the 18th International Conference on Natural Language Processing, pages 583–592
Silchar, India. December 16 - 19, 2021.

Dataset	Type	Examples of Tasks
Resumes	Verbal	We compared the techniques with which **[low power cascade amplifiers can be developed]**.
	Nominal	Verified the credibility of GA algorithm in **[designing of orthogonal waveform for MIMO radar]**.
TechCrunch	Verbal	But Kaliszan and Robertson realized that commercial security was so backward that just **[implementing the established principles of machine vision]** and the cloud could create a huge company.
	Nominal	Remote work is what led to **[the development of GitLab's publicly viewable handbook]**.
Patents	Verbal	Particularly, **[a new e-mail message is created for each software package to be deployed]** .
	Nominal	The computer implemented method includes performing, using a processor, **[a static timing analysis of the integrated circuit]**.
Reuters News	Verbal	At noon the bank had **[estimated the shortfall at 500 mln stg]**.
	Nominal	Stock market analysts said today's generally weak stock market plus unwinding of positions after **[heavy buying of BAT shares in the run-up to the results]** caused the fall in the share price.

Table 1: Examples of Tasks mentioned in various datasets. The task phrases are enclosed in square brackets and are highlighted in bold within sentences. "Type" indicates the syntactic type of the task phrase which depends on the POS tag of the head word of the phrase – verbal (e.g., `developed`) or nominal (e.g., `development`).

ited to a very few tasks. Using tasks extracted from a large corpus, we can automatically augment such common sense knowledge.

In this paper, we focus on the problem of automatic extraction of tasks from text and propose two techniques for that. The first technique makes use of linguistic patterns and resources such as WordNet (Miller, 1995). The second technique uses a weakly supervised BERT-based neural model and employs the Snorkel framework (Ratner et al., 2017) for automatically creating a labeled training dataset. The rest of the paper is organized as follows – Section 2 discusses relevant past work. Section 3 describes the two proposed techniques for the automatic extraction of tasks from text. Section 4 describes detailed experimental analysis including dataset descriptions, baselines, and evaluation. Finally, We conclude in Section 5 along with some potential future work.

2 Related Work

To the best of our knowledge, ours is the first attempt to introduce the problem of task extraction and propose extraction techniques for it. Event extraction (Xiang and Wang, 2019) is a related but different problem as compared to task extraction. Definition of an *event* varies depending on its application and context. Sims et al. (Sims et al., 2019) have defined an *event* to be what is depicted as actually occurring in text (also referred as *realis events*). Such events are expected to have four important aspects (non-negation, tense, genericity, and modality) which we compare and contrast with tasks in the top four rows of Table 2. We also describe two more important aspects of tasks – *volitionality* and *need for expertise*. Overall, although there is overlap between tasks and events, neither is an subset of the other.

3 Task Extraction

In this section, we discuss the two techniques for extraction of tasks from text.

3.1 Using Linguistic Patterns

Linguistic patterns for task extraction do not need any training data. We define an *action noun* as a noun that indicates an action, activity etc.; e.g., `improvement`, `design`, `review`, `selection`, `administration`. A noun is accepted as an action noun if (i) its hypernym tree (for any of its top $k_0 = 2$ senses) includes *action indicator words* like `work`, `activity` , `human_action`, `group_action` etc. (e.g., hypernym tree for sense 2 of `improvement` is: *change of state* $\rightarrow$ *change* $\rightarrow$ *action* $\rightarrow$ ***human_action***); or (ii) Alternatively, the noun's category in WordNet should be `noun.act`. Generally, we do not consider abstract nouns (e.g., `idea`) as action nouns; and (iii) the noun is not present in a domain-specific negative list; for IT domain, some examples of negative action nouns are: `project`, `technology`, `job`, `approach`, `practice`, `procedure` etc.

Similarly, we accept a verb V to be an action verb[1] if: (i) one of the derivationally re-

[1]The list of action nouns and verbs which is created using WordNet, will be made available upon request.

Aspect	Description
Non-negation	Events should be explicitly mentioned as having occurred; typically there is no direct negation used to describe an event. Tasks are also similar to events in this aspect. In the sentence `He did not implement the solution`, neither an event nor a task is mentioned.
Tense	Events must be in the past or present tense. Tasks need not only be in past or present tense, they can be mentioned in future tense as well. In the sentence `He will implement the solution`, a task `implement the solution` is mentioned but it is not an event.
Genericity	Generic events (`Engineers build bridges`) describe a general category as against specific events which describe a specific occurrence (`L&T engineers built this bridge`). In event extraction, only specific events are considered whereas tasks can be generic events.
Modality	Only *realis* events which have *actually occurred* are considered as events. All other modalities such as *belief*, *hypothesis*, *desire* are not considered events but these can be considered as tasks. In the sentence `Engineers are supposed to build bridges which last for years`, a task `build bridges` is mentioned but it is not an event.
Volitionality	Tasks are those actions which are carried out by the actor *volitionally*. For example, `The bridge collapsed` is an event but it is not a task because the action of `collapsing` is not carried out volitionally by any actor.
Expertise	Unlike events, we define tasks as those actions which need some domain expertise or knowledge by the actor for execution. For example, `John entered the restaurant` is an event but not a task.

Table 2: Comparing *Events* and *Tasks* based on various aspects

lated nominal forms of V is an action noun; or (ii) the verb category in WordNet is any of: `verb.change`, `verb.motion`, `verb.creation`, `verb.social`, `verb.communication` etc. For example, there are 2 nouns related to sense 1 of the verb `provide`, namely, `provision` and `provider`. The second sense of `provision` includes `activity`, making it an action noun and hence `provide` is an action verb. Other examples: `improve`, `stabilize`, `plan`, `control`, etc.

Examples of linguistic rules to identify tasks having different syntactic structures are as follows:
• A noun compound (i.e., a sequence of nouns) is a task if the last noun is an action noun. Examples: `process improvement`, `user interface design`, `version control`, `asset management`
• An action verb in simple present/past tense or in gerund form and its direct object noun phrase (NP) form a task. Examples: `manage data center`, `implement quality processes`, `managing attrition`, `maintaining financial discipline`, `resolved customer complaints`
• An action verb V in gerund form connected to a preposition p using dependency relation (DR) *prep* followed by an NP connected to p using DR *pobj* is a task. Examples: `managing of large teams for customer support`, `coordinating with various vendors`
• Same as above, except instead of V a noun compound headed by an action noun is required. Examples: `analysis of existing bugs`, `Eigenvalue computation by application of numerical computation techniques`

3.2 Weakly Supervised BERT-based Task Extraction

The linguistic patterns based approach has certain limitations which need to be addressed to further improve the task extraction accuracy. The patterns check for the presence of action verbs and nouns and then extract their entire verb or noun phrases as tasks. However, the presence of action verbs or nouns is just a *necessary* condition and not a *sufficient* condition for being tasks. Two important aspects of tasks *volitionality* and *need for expertise* are not checked explicitly. Moreover, there is a challenge of polysemy which is not handled explicitly. A verb (or noun) may be an action verb (or an action noun) in one particular sense but may not be an action verb (or action noun) in another sense. For example, `synthetic data generation` is a valid task but `next generation of chip technology` is not a valid task because of different senses of the noun `generation`.

To overcome the above-mentioned limitations of our linguistic patterns based approach, we propose to learn a classification model which predicts whether any noun or verb in a sentence represents a *head word* of a valid task phrase. Linguistically, the head word of a phrase is the word which determines the syntactic category (e.g., noun phrase, verb phrase) of the phrase. We use the following definition of a head word considering the dependency parse tree – the head word of a phrase is syntactically the most important word in the phrase which connects it to the rest of the sentence, all other words in the phrase are directly or indirectly dependent on the head word. Once we identify

head words of task phrases, we use the dependency tree structure of the sentence to get the corresponding complete task phrase. The rules for phrase expansion are described later in detail.

3.2.1 Classification Problem

Input: A word w in sentence S
Output: Predict one of the two class labels - TASK and NOT-TASK indicating whether or not the word w is a head word of a valid task phrase

3.2.2 Training Data

As there is no prior work for addressing this problem of extracting tasks, there are no readily available annotated datasets which we can use for training the above-mentioned classification model. Hence, we use the Snorkel framework (Ratner et al., 2017) for rapidly and automatically creating labeled training data. Snorkel enables writing multiple *labeling functions* (LFs) where each LF expresses an arbitrary heuristic for class label predictions. These LFs can have unknown accuracies and correlations but Snorkel denoises their outputs and combines their predictions to arrive at a final probability distribution over labels for each instance. A large training set can then be constructed rapidly using these automatically assigned *soft* labels (because of a probability distribution over labels and not a single hard label for each instance) and this training data can be used to train a machine learning model. We designed several LFs that capture various linguistic characteristics which we expect to be present in tasks.

3.2.3 Labeling Functions

We designed the following LFs where each LF assigns TASK or NOT-TASK for a classification instance. An LF need not assign labels for all instances, it may ABSTAIN for certain instances where it is unsure. A classification instance is a combination of a word w, the corresponding sentence S, and the dependency tree DT of S.

Action verbs or nouns: If the word w is not an action verb or noun (as per the list of action verbs and nouns prepared using WordNet in Section 3.1) then it is NOT-TASK. Here, the sentence context is not used and the decision is only based on the word w. E.g., nouns such as `book`, `culture` and verbs such as `situate`, `lack` are not tasks. All the subsequent LFs also predict NOT-TASK for non-action verbs and nouns but they also predict TASK or NOT-TASK for action nouns and verbs provided certain other conditions are satisfied.

Negation modifier: If the word w is modified by any negation indicating word (e.g., `not`, `never`) through dependency relation neg in DT then it is NOT-TASK. E.g., `They did not develop any weather prediction model`[2]. We also consider other ways of expressing negation such as `failed to develop` or `absence of development`.

Animate or organization agent: If the agent (dependency child with relation $nsubj$ or $agent$) of the verb w is *animate* or corresponds to some organization, then it is TASK. This LF captures volitionality in an implicit way as the animate agents (or organizations) indicate that the action corresponding to verb w is likely to be carried out volitionally. Here, animate/organization agents are those words which are – i) personal pronouns like `he`, `she`, `we`, ii) named entities of type PERSON or ORGANIZATION, or iii) person or organization indicating common nouns like `engineer`, `farmer`, `department` (these words have `person` or `organization` as their ancestors in WordNet hypernym tree). E.g., `Any overseas data demands are screened by the` **`department`**`.`

Inanimate agent: If the agent of the verb w is *inanimate*, then it is NOT-TASK. This LF captures the opposite characteristics as compared to the previous LF. The heuristic is that if any action is carried by an inanimate agent then it is unlikely to be a task. Here, inanimate agents are those words which are – i) event indicating nouns (e.g., `storm`, `pandemic`) or ii) natural objects or substances (e.g., `stone`, `water`). Again, lists of such words are created using WordNet hypernym structure. E.g., `The coronavirus` **`pandemic`** `accelerated the shift to e-commerce.`

Volition marker: If the verb w is modified by an adverb (dependency child with relation $advmod$) explicitly indicating volition, then it is TASK. Examples of volition indicating adverbs are `deliberately`, `voluntarily`, `intentionally`. E.g., `He` **`voluntarily`** `engages in self-developmental activities.`

Non-volition marker: If the verb w is modified by an adverb explicitly indicating non-volition, then it is NOT-TASK. Examples of non-volition indicating adverbs are `accidentally`, `unintentionally`. E.g., `He` **`accidentally`** `pressed the send button.`

[2]The word w is underlined and the same convention is followed for subsequent examples

Explicit expertise marker: If the word w occurs in explicit *expertise indicating context* in the dependency tree DT, then it is TASK. One of the key aspect of a task is that it needs certain domain expertise to be executed. Expertise indicating context can be – i) w being modified by an adjectival clause headed by `using` or `leveraging`, or ii) w being preposition phrase modifying nouns/verbs such as `knowledge of` or `expertise in`. E.g., `You can dynamically deliver language files to your mobile apps` **`using`** `SDKs.`

Expertise score using corpus statistics: If the word w has high *expertise score* based on corpus statistics then it is `TASK`. This LF does not use the sentence context for the word w. Here, we compute expertise score for each action noun and verb using statistics from a large corpus; and then choose top 100 action verbs and nouns using this expertise score. We used 3.6 million sentences from ukWaC, a very large web-derived corpus of English (Ferraresi et al., 2008). For each action verb and noun, the expertise score is computed as follows:

$$ExpScore(w) = log(N^e_w + 1) \times \frac{N^e_w}{N_w} \quad (1)$$

Where, N_w: No. of times the word w appears in the corpus and N^e_w: No. of times the word w appears in the explicit *expertise indicating context* (as described in the previous LF). The score for a word will be high if both of the following conditions are true – i) the conditional probability of observing it in expertise indicating context is high, and ii) the absolute frequency with which it appears in expertise indicating context is high. This is motivated by the patterns scoring formula used by Thelen and Riloff (2002). E.g., `The Fosters will develop media like podcasts and videos.` and `The work involves experimental investigation of droplet impingement over a heated surface.`

Presence of direct object: If the word w has a direct object ($dobj$) or a passive subject ($nsubjpass$) then it is `TASK`. For actions expressed using nouns, prepositional phrase headed by `of` is considered similar to a direct object (`implemented the solution` ⇒ `implementation of the solution`). Here, the heuristic is that the presence of direct object (or passive subject for passive voice verbs) for an action verb increases likelihood of it being a more meaningful task. E.g., `It recently published a` **`handbook`**`.`; `The` **`post`** `was restricted on social media.`; and `It asks user for selection of a particular` **`webpage`**`.`

Absence of direct object or prepositional modifier: If the verb w does not have any direct object, passive subject, or any prepositional modifier, then it is NOT-TASK. This LF captures the opposite characteristic as compared to the previous LF, with the heuristic that such verbs are unlikely to constitute meaningful tasks. E.g., `It allows the VM to begin operating quickly.`

Adjectival clause modifier: If the verb w is a head word of an adjectival clause modifying some noun, then it is `NOT-TASK`. Here, the heuristic is that such verbs simply provide extra information about a noun and are unlikely to be tasks. E.g., `It left thousands of sensitive health records exposed to the internet.`

Compound noun modifier: If the noun w modifies another noun as a compound modifier, then it is `NOT-TASK`. E.g., `Rivian sets a delivery date.`

Number-like modifier: If the noun w is modified by a number, an ordinal, a cardinal, or a number-like modifier like `next`, then it is `NOT-TASK`. E.g., `The solutions that the first generation of clean tech investors backed were economically unfeasible.`

3.2.4 BERT-based classification model

We propose a BERT-based (Devlin et al., 2018) classification model which predicts an appropriate class label (TASK vs NOT-TASK) for each word in a sentence. The annotated data needed for training this model is created automatically using the Snorkel framework with the labeling functions described above. Each instance is annotated with soft labels, i.e., a probability distribution $y_{gold} \in \mathbb{R}^2$ over TASK and NOT-TASK. Each instance is a combination of a word w, its POS tag p, and the complete sentence S.

We now describe the classification model in detail. Figure 1 depicts the model architecture. First, embedded representation $x_w \in \mathbb{R}^{768}$ is obtained for the word w using a pre-trained BERT transformer model.

$$x_w = BERT(S, w) \quad (2)$$

We then use a linear feed-forward layer to get a more compressed representation $x'_w \in \mathbb{R}^{10}$ of the word.

$$x'_w = ReLU(Wx + b) \quad (3)$$

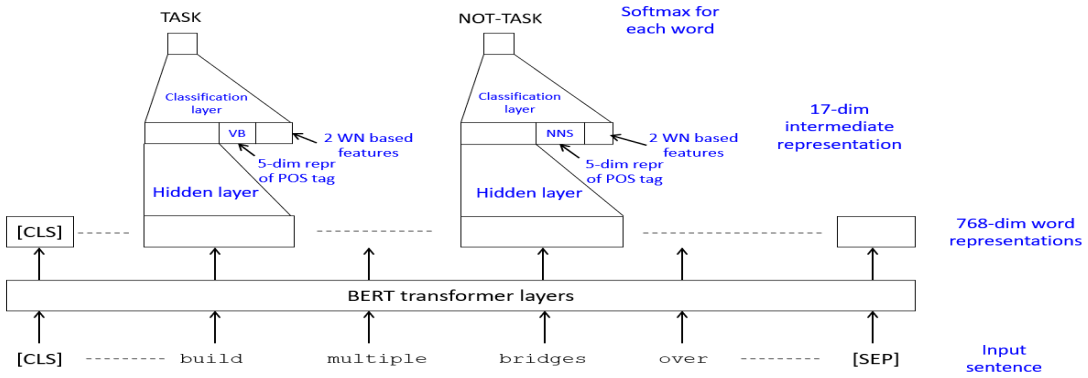

Figure 1: Architecture of BERT-based Task Extraction model

Here, $W \in \mathbb{R}^{10\times768}$ and $b \in \mathbb{R}^{10}$ are learnable weights. We compress the word representation to lower dimensions because we use 3 additional features described below and we do not want the BERT-based features to overwhelm these additional features.

POS tag: Part of speech tag of the word w is a key feature because the head word of a task can only be a noun or verb. We use embedded representation $x_p \in \mathbb{R}^5$ of the POS tag of w using an Embedding layer with learnable weights ($\mathbb{R}^{N_p\times5}$ where N_p is the number of distinct POS tags).

WordNet-based features: We hypothesize that more *specific* words are more likely to be task head words than *generic* words. Hence, we use two WordNet-based features to implicitly estimate *specificity* of the word w – i) Hypernym depth and ii) Corpus frequency. Hypernym depth of a word is the number of levels in the hypernym tree between that word and the root (`entity` in case of all nouns). The higher the hypernym depth, the higher is its specificity. E.g., hypernym depth of `diagnosis` is 8 which is higher as compared to a more generic word `action` with hypernym depth of 5. Similarly, lower the corpus frequency of a word, higher is its specificity. We use the corpus frequencies provided for each lemma in each synset in WordNet (Jurafsky and Martin, 2021). E.g., corpus frequency of `see` is 613 but for `analyze` it is only 21 which is more specific word. For each word, we use WordNet-based features vector $x_{wn} \in \mathbb{R}^2$

Overall representation of the word w, $h_w \in \mathbb{R}^{17}$ is now concatenation of x'_w, x_p, and x_{wn}. This is then passed through the final classification layer to get the predicted label distribution $y_{pred} \in \mathbb{R}^2$.

$$h_w = Concatenate([x'_w; x_p; x_{wn}]) \quad (4)$$

$$y_{pred} = Softmax(W'h_w + b') \quad (5)$$

$$loss = KLDLoss(y_{gold}, y_{pred}) \quad (6)$$

Here, $W' \in \mathbb{R}^{2\times17}$ and $b' \in \mathbb{R}^2$ are learnable weights. The predicted label distribution is then compared with the gold standard or expected distribution y_{gold} to compute KL divergence loss which is back-propagated during the training process. We also fine-tune the final encoder layer of BERT.

3.2.5 Phrase Expansion

The classification model identifies task head words which need to be expanded to get complete task phrases. We use a few simple rules to expand head words to phrases using the dependency tree of the sentence. Basically, we need to get the phrase corresponding to the dependency subtree rooted at the head word but we need to discard certain dependencies. We recursively collect a set of dependency children starting from the head word and construct the phrase from the leftmost child to the rightmost child. However, we do not consider dependency child connected to its parent with certain dependency relations and hence do not recurse on such children further. Dependency relations which we discard are – i) $nsubj$, $agent$ (because task phrase does not contain the agent who executed the task); ii) $relcl$, $advcl$, $ccomp$, $appos$ (to avoid getting complete dependent clauses or appositives describing extra information inside a task phrase); iii) aux, $auxpass$ (for not including auxiliary verbs in task phrases). E.g., consider the task head word `analyzed` from the sentence in Figure 2. Here, the expanded task phrase is `analyzed traffic at internet exchanges`. Here, `firm` is excluded because its dependency relation $nsubj$ is discarded, but other dependency children of `analyzed` are included recursively.

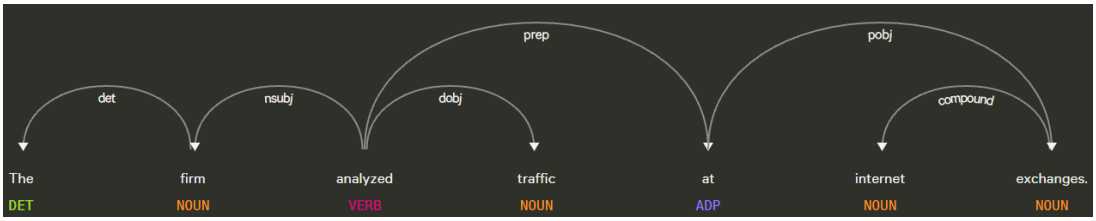

Figure 2: Example of a dependency tree and phrase expansion for the task head word `analyzed`. (Courtesy: spaCy dependency visualizer at `https://explosion.ai/demos/displacy`)

4 Experimental Analysis

In this section, we describe our experiments in detail including datasets, baselines, evaluation metrics and results.

4.1 Datasets

In order to evaluate the task extraction performance, we chose 4 datasets from different domains.

• **Resumes**[3]: Set of resumes of candidates shared with our organization

• **TechCrunch**[4]: Articles from technical domain published on TechCrunch in 2020

• **Reuters**: A collection of 10,788 documents from the Reuters financial newswire service from the well-known Reuters-21578 corpus (Lewis, 1997).

• **Patents**[5]: Abstracts of patents assigned to IBM in the years 2000-2019 which are scraped from Google Patents

4.1.1 Training Data

The training data for our BERT-based task extraction model is generated automatically using the Snorkel framework based on several labeling functions described in Section 3.2.3. We randomly chose 1000 documents from the 4 datasets (250 documents from each) – Resumes, TechCrunch, Reuters, and Patents. The dataset consists of 19268 sentences where 98044 words (verbs and nouns) are assigned soft labels using Snorkel's labeling model (Ratner et al., 2017) which combines predictions of our labeling functions. Out of 98044 words, 21892 words were labeled as likely TASK head words, i.e., they were assigned TASK probability greater than 0.5. For all the remaining words in these sentences, NOT-TASK is considered as a hard label. Table 3 shows various statistics of these labeling functions on this training data. Detailed hyper-parameter details used for training this model are inlcuded in the Appendix.

Labeling function	Cov	Overlap	Conflict
non action nouns/verbs	0.625	0.625	0.000
negation modifier	0.630	0.629	0.006
animate/org agent	0.680	0.667	0.021
non-animate agent	0.638	0.634	0.010
volition marker	0.625	0.625	0.000
non-volition marker	0.625	0.625	0.000
explicit expertise marker	0.629	0.629	0.001
corpus expertise score	0.673	0.660	0.013
direct object	0.799	0.687	0.013
adjectival clause	0.641	0.630	0.002
no object or pp	0.708	0.651	0.029
compound modifier	0.641	0.625	0.000
number-like modifier	0.625	0.625	0.000

Table 3: Analysis of labeling functions over our training dataset (Cov: Fraction of training instances labeled (not abstained) by the LF, Overlap: Fraction of training instances where the LF has predicted along with at least one other LF, Conflict: Percentage of overlapping training instances where there is mismatch of predicted label with at least one other LF)

Dataset	#Sentences	#Tasks
Resumes	1297	167
TechCrunch	292	251
Reuters	178	89
Patents	102	100
Total	1869	607

Table 4: Details of the evaluation dataset

4.1.2 Ground Truth

In order to create ground truth for evaluating various task extraction techniques, we manually annotated 20 documents from each of the 4 datasets with gold-standard task head words and complete task phrases. This dataset[6] consists of 1869 sentences where 607 tasks are annotated (see Table 4).

[3]This is an internal dataset and can not be made public due to privacy reasons.

[4]`https://www.kaggle.com/sumantindurkhya/techarticles2020`

[5]`https://www.kaggle.com/federicolusiani/ibm-patents`

[6]Evaluation dataset as well as automatically labeled training dataset (excluding Resumes) will be made available upon request.

Dataset	Technique	Only Task head word			Lenient evaluation			Strict evaluation		
		P	R	F1	P	R	F1	P	R	F1
Resumes	EvExtB1	0.553	0.166	0.255	0.380	0.103	0.162	0.314	0.086	0.136
	EvExtB2	0.335	0.669	0.447	0.373	0.714	0.49	0.232	0.454	0.307
	Linguistic Patterns	0.582	0.771	**0.663**	0.551	0.730	**0.628**	0.429	0.568	**0.488**
	BERT Extractor	0.552	0.675	0.607	0.505	0.589	0.544	0.311	0.368	0.337
TechCrunch	EvExtB1	0.354	0.222	0.273	0.343	0.229	0.274	0.217	0.144	0.173
	EvExtB2	0.312	0.763	0.442	0.294	0.734	0.419	0.187	0.476	0.268
	Linguistic Patterns	0.404	0.510	0.451	0.420	0.542	0.473	0.239	0.310	0.270
	BERT Extractor	0.449	0.732	**0.556**	0.422	0.694	**0.524**	0.262	0.439	**0.328**
Reuters	EvExtB1	0.323	0.370	0.345	0.294	0.364	0.325	0.139	0.170	0.153
	EvExtB2	0.188	0.716	0.297	0.188	0.761	0.302	0.095	0.386	0.152
	Linguistic Patterns	0.210	0.358	0.265	0.218	0.364	0.272	0.122	0.205	0.153
	BERT Extractor	0.314	0.716	**0.436**	0.296	0.682	**0.412**	0.161	0.375	**0.225**
Patents	EvExtB1	0.533	0.075	0.132	0.556	0.085	0.148	0.267	0.034	0.061
	EvExtB2	0.371	0.774	0.502	0.370	0.752	0.496	0.179	0.385	0.244
	Linguistic Patterns	0.420	0.472	0.444	0.515	0.590	0.550	0.220	0.248	0.233
	BERT Extractor	0.524	0.830	**0.642**	0.522	0.803	**0.633**	0.268	0.419	**0.327**
Average	EvExtB1	0.441	0.208	0.251	0.393	0.195	0.227	0.234	0.109	0.131
	EvExtB2	0.302	0.731	0.422	0.306	0.740	0.427	0.173	0.425	0.243
	Linguistic Patterns	0.404	0.528	0.456	0.426	0.557	0.481	0.253	0.333	0.286
	BERT Extractor	0.460	0.738	**0.560**	0.436	0.692	**0.528**	0.251	0.400	**0.304**

Table 5: Comparative task extraction performance of our proposed techniques Linguistic Patterns and Weakly supervised BERT-based Task Extractor

4.2 Baselines

We consider two recent event extraction techniques as baselines for comparing the performance of our task extraction techniques.

EvExtB1: The first baseline is literary event extraction technique proposed by Sim et al. (2019). It is trained on literature dataset using a BiLSTM based model which used BERT token representations.

EvExtB2: The second baseline is an Open Domain Event Extraction technique proposed by Araki and Mitamura (2018). This is a more competent baseline because the events are not restricted to a domain or a syntactic type. It uses a BiLSTM based supervised event detection model which is trained on distantly generated training data.

For both the baselines, we use pre-trained models provided by the authors. Both the baselines identify event triggers that are considered as task head words and complete task phrases are identified using the phrase expansion rules described in Section 3.2.5.

4.3 Evaluation Metrics

Any gold-standard task phrase is counted as a true positive (TP) if there is a "matching" predicted task, otherwise it is counted as a false negative (FN). Here, two task phrases are considered to be "matching" if there is at least 80% string similarity between them for *strict evaluation* and 50% for *lenient evaluation*. All the remaining predicted tasks which are not TPs, are counted as false positives (FP). In addition, similar to event triggers, we also compute TPs, FPs and FNs considering only task head words. Precision, recall and F1-score are then computed for each of these three evaluation strategies – strict evaluation, lenient evaluation and considering only task head words.

$$P = \frac{TP}{TP + FP}; R = \frac{TP}{TP + FN}; F1 = \frac{2 \cdot P \cdot R}{P + R} \quad (7)$$

4.4 Results

We evaluate our proposed techniques – linguistic patterns and weakly supervised BERT-based task extractor, on 4 different datasets using 3 evaluation strategies. We compare the performance of the proposed techniques with two event extraction baselines. Table 5 shows the detailed results. Except the Resumes dataset, the BERT-based task extractor outperforms all other techniques on all the datasets. Considering the macro-average across datasets, the BERT-based task extractor turns out to be the best overall technique which also performs consistently across datasets. We also carried out ablation analysis to evaluate contribution of POS tag and WordNet-based features and observed that these features have minor positive contribution. Table 6 shows a comprehensive list of tasks extracted by the BERT-based task extractor from the 4 datasets.

Resumes:
-weld the P91 steel plate
-reduce the surface area of radiators
-develop a FEM code of multidimensional small deformation plasticity
-Cold flow analysis of double ramp flame holder
-Designed and analyzed a two stage worm gear box
-facilitate cab services
-Implemented a dynamic memory allocator for C
-Comparison of Different Clustering Techniques
-the generation of layered NAND gate
-Development of desktop application using deep learning
TechCrunch:
-enhance and improve antitrust regulations on the platform economy
-background data - mining of internet users
-tackling abusive behavior
-ensuring fairness in digital marketplaces
-verifying any system weaknesses and functioning of devices
-load up prior versions of iOS
-fire up a simulated iPhone and hunt for potential bugs
-discover potential security bugs
-spin up a virtualized ARM device (including iOS devices) in a browser
-filed a lawsuit against the virtualization software company
Reuters:
-facilitate a transaction
-fixed the value of the new Cruzado currency
-studying financially superior alternatives
-A bill also was introduced
-monitor coffee imports
-control the spread of the disease
-the susceptible clones would be replaced
-the disease was detected in nurseries
-restore normal moisture to the cane
-made appropriate declarations at customs points
Patents:
-a log transfer to the standby machine is performed
-the executing program is blocked
-the spatial index may be partitioned
-partition a spatial index into a plurality of portions
-A set of congestion cost corresponding to the set of pattern routes is computed
-data processing
-verification of a digital cir cuit design
-disseminate the write request
-performing multimodal analysis on the multimedia stream
-maintain the PIN diode bias as high as possible

Table 6: Examples of Tasks extracted from various datasets

4.5 Implementation Details

In this section, we describe the hyper-parameters used for training our BERT-based Tasks Extraction model. We have not carried out extensive hyper-parameter tuning but we set aside a small subset of training set as validation set, tried a few settings and chose the best one. We then re-trained our model using the entire training set with this set of hyper-parameters: batch size = 16 sentences, maximum sentence length = 128 tokens, number of epochs = 2, Adam optimizer with learning rate = 0.001. For avoiding overfitting, we used a dropout of 0.4 probability over the 768 dimensional token representation output by BERT. We also used gradient clipping to keep maximum norm of the gradient vector below 5. We used pre-trained base model of BERT from HuggingFace: `bert-base-uncased`[7]. While training our model, we also fine-tuned the last encoder layer of BERT (`encoder.layer.11`) and kept other layers' parameters frozen.

5 Conclusion and Future Work

In this paper, we have introduced a new NLP task of automatic extraction of *tasks* from text, highlighted the motivation behind it, and its potential applications. We described various aspects of tasks and highlighted how they compare with another popular NLP task of event extraction. We proposed two techniques for task extraction – i) linguistic patterns and ii) BERT-based weakly supervised neural model. We demonstrated effectiveness of our techniques on 4 datasets from different domains and compared them with other competent baselines. Given that ours is the first attempt for extracting tasks from text and the approach is only weakly supervised and does not demand any heavy manual annotation efforts, the overall performance is encouraging. However, there is still scope for the improvement which we plan to pursue as a future work. Also, we wish to refine the labeling functions to better capture linguistic characteristics of tasks such as volitionality, need for expertise, and execution within a small time period.

Acknowledgments

We thank the anonymous reviewers for their comments. We also thank our colleagues Nitin Ramrakhiyani and Dr. Swapnil Hingmire for helping to set up the event extraction baselines.

References

Jun Araki and Teruko Mitamura. 2018. Open-Domain Event Detection using Distant Supervision. In *Proceedings of the 27th International Conference on*

[7] https://huggingface.co/bert-base-uncased

Computational Linguistics (COLING), pages 878–891, Santa Fe, NM, USA.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukwac, a very large web-derived corpus of english. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google*, pages 47–54.

Dan Jurafsky and James Martin. 2021. *Speech & language processing*. https://web.stanford.edu/ jurafsky/slp3/.

David Lewis. 1997. Reuters-21578 text categorization test collection, distribution 1.0.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access.

Matthew Sims, Jong Ho Park, and David Bamman. 2019. Literary event detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3623–3634.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Michael Thelen and Ellen Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of the 2002 conference on empirical methods in natural language processing (EMNLP 2002)*, pages 214–221.

Wei Xiang and Bang Wang. 2019. A survey of event extraction from text. *IEEE Access*, 7:173111–173137.

A Resources created using WordNet and ukWaC corpus

As described in Sections 3.1 and 3.2.3, we have created several key resources using WordNet hypernym structure which includes lists of – action verbs, action nouns, person indicating nouns, organization indicating nouns, natural objects, and substances. We also computed expertise scores for various action nouns and verbs using a subset of ukWaC corpus. In our labeling function, we use top 100 action nouns and verbs as per this score. These lists which are created using WordNet and ukWaC corpus, will be made available upon request.