Dynamic Event Type Recognition and Tagging for Data-driven Insights in Law-Enforcement

Shayan Zamanirad · Boualem Benatallah · Moshe Chai Barukh · Carlos Rodriguez · Reza Nouri

Abstract In Law Enforcement, investigators are typically tasked with analyzing large collections of evidences in order to identify and extract key information to support investigation cases. In this context, events are key elements that help understanding and reconstructing what happened from the collection of evidence items. With the ever increasing amount of data (e.g., e-mails and content from social media) gathered today as part of investigation tasks (in most part done manually), managing such amount of data can be challenging and prone to missing important details that could be of high relevance to a case. In this paper, we aim to facilitate the work of investigators through a framework for deriving insights from data. We focus on the auto-recognition and dynamic tagging of event types (e.g., phone calls) from (textual) evidence items, and propose a framework to facilitate these tasks and provide support for insights and discovery. The experimental results obtained by applying our approach to a real, legal dataset demonstrate the feasibility of our proposal by achieving good performance in the task of automatically recognizing and tagging event types of interest.

Keywords Event Recognition · Event Tagging · Data-driven Insights · Law Enforcement

1 Introduction

Law enforcement has long benefited from the advances in Information and Communication Technologies (ICT) including information and knowledge management as well as leveraging big data analytics and intelligence. Techniques stemming from social network analysis [1], data mining [18], machine learning [47] and video analytics [24] have pushed the frontiers of the ability to process case-related information as part of law enforcement investigations. However, the ever increasing amount of information gathered

S. Zamanirad, B. Benatallah, M. Chai Barukh, C. Rodriguez (✉)
UNSW Sydney, NSW 2052, Australia
Tel.: +61-2-9385-5518
E-mail: {shayanz,boualem,mosheb}@cse.unsw.edu.au, carlos.rodriguez@uc.edu.py

*R. Nouri
QANTAS Airways, NSW 2020, Australia
E-mail: s.nouri@unswalumni.com
*This work was done while the author was at UNSW Sydney

This is a post-peer-review, pre-copyedit version of an article published in Computing.
The final authenticated version is available online at: https://doi.org/10.1007/s00607-020-00791-z

today as part of investigation tasks poses an important challenge in terms of being able to navigate and understand information that may prove crucial in a given case.

As a real-life example, an investigator may be tasked to retrieve the passenger manifest for all flights between two cities over a given time window and then check for evidence that a person of interest did indeed travel between these cities on a certain date. Similarly for bank transactions, witness statements or telephone records. Ultimately, each piece of data needed to prove an offense is obtained, and then manually analyzed by the investigator for significance to the case. Clearly, as the number of such tasks build up, and relevant information is added to the case, the cognitive load for investigators increases [40][4]. This is further exacerbated in today's technological landscape, where information may be obtained from multiple, disparate sources (e.g., emails, phone calls, SMSs and social media).

The primary objective of an investigation is to uncover sufficient evidence from various information sources (e.g, witness statements, emails, documents, collaboration messages, social media posts, bank transactions). In our work we focus on textual and unstructured information items (e.g., emails, SMSs, PDF files) containing events that can be relevant in the context of an investigation. *Events* are key elements for understanding and reconstructing what has happened from a collection of evidence items. Events are categorized by types. For example, the event-type *Phone Call* represents all sorts of voice communications that happen through traditional landlines, cellular phones and communication software applications. Another example is the event-type *Bank Transaction*, which represents all transactions performed having a bank as an intermediary, including ATM operations, check cashing and transfers between bank accounts. The identification of individual events is not only one of the first step toward understanding the case but it also facilitates the reconstruction of chains of hypotheses and facts to help understand what happened in the case, identification of parties involved, understanding of its temporal dynamics, among other aspects [36]. However, identifying events from unstructured information items is a hard challenge due the inherent ambiguity in natural language (e.g., various mentions of a same entity, jargons, misspellings).

In this paper, we aim to facilitate the work of investigators with a framework and techniques for deriving insights from data. Inspired by word embeddings and their impact on various application domains [7], we develop a novel vector space model and techniques to represent *event-types*. While basic word embedding only considers "words" in a vector, we focus on novel extensions to represent specific constituents of events. More specifically, we focus on leveraging event embedding techniques for the auto-recognition and dynamic tagging of event-types from evidence logs, which can facilitate the investigation process. This work makes the following main contributions:

1. We devise a framework for data-driven insights: From generating *training data*, *processing of data into contextualized knowledge*, to *techniques for insights and discovery*. A key component we propose is the notion of *dynamic tags*, i.e., tags that are automatically created via machine processing. The use of tags provides a powerful mechanism for the organization, navigation, summarization and understanding of large amounts of data [44].
2. We leverage techniques from Natural Language Processing (NLP) and word embeddings [31] to enable the recognition of events from case-related corpora. Word embeddings is a technique that investigates how linguistic elements (words) are semantically similar based on their distributional properties [22]. Extrapolating upon this idea, we encode event-types as vectors based on the distributional semantics of the evidence text. We then leverage on the similarity properties of the vectors to help identify when sentences relate to a particular event-type. Our framework also includes methods for ongoing and incremental feedback from domain experts.

The rest of the paper is organized as follows: In Section 2 we describe data (i.e., evidence) collection and annotation in the typical workspace of an investigator. Section 3 presents our proposed approach to dynamic recognition and tagging of event-types. Here, we first discuss our overall framework and then discuss each of its component layers, namely, training data generation, event-type recognition, and insights and discovery. We then validate our proposed approach in Section 4, followed by a discussion in Section 5. Related work is presented in Section 6, and finally we provide our concluding remarks and future work in Section 7.

2 Evidence Collection & Analysis

Data collected (in this case *evidence*) during an investigation provide fundamental information to help understand a case and build legal arguments in a court-of-law [28]. Evidences can originate from different sources, including interviews transcripts, surveillance (e.g., security cameras), telecommunication channels (e.g., SMS messages), witness statements, investigation materials (e.g., habitation check report) and electronic devices (e.g., laptops). Table 1 shows the different sources from which evidences are collected during an investigation.

Table 1 Examples of Evidence Sources

Evidence sources	**Examples**
Interview	Video, audio and transcript
Surveillance	Video, audio, photograph, tracking, notes, and runsheet/log
Telecommunication	Transcript, audio, SMS, call log and e-mail
Witness statement	Police and civilian narratives, Border Force Custom, Border Force Immigration, translator, and forensics
Investigation material	Warrant, evidentiary certificate, photograph, company search, habitation search, notes, and forensic item
Electronic device	Mobile phone, computer, tablet SIM card, portable storage and GPS

Typically, a case handles a large amount of evidences that are collected and stored for further analysis [4]. We refer to the collection of evidence as the *evidence log*. Relevant aspects related to managing such evidence logs in the context of police investigations include the ability to quickly search for evidences and identify how they are connected to the various events, people, location, objects and offenses involved in the investigation.

Figure 1 shows an illustration of an investigator's typical workspace. We also show how evidences are typically annotated to add semantics to data. Moreover, we exemplify how relationships can be *inferred* between two items that would otherwise remain unconnected. In Figure 1, we have a *witness statement* annotated by the investigator with two event-types, firstly a **travel movement** of Andrew who went to meet Phillip; and then a **bank transaction** between Phillip and Andrew. Next, we have an email exchange between Andrew and another person, Norman. The name of a **Person** as *named entities* [30], as well as **timestamps** are also annotated. Third we have a video surveillance that recorded Norman stealing a car.

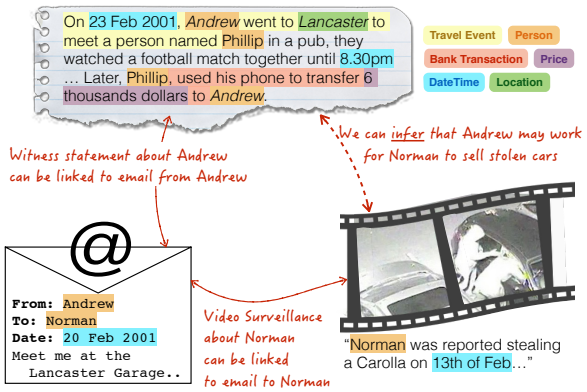

Fig. 1 Illustration of an investigator's workspace.

Beyond evidence annotation for summarization and search purposes, the resulting annotations can be exploited to infer further, useful insights. For example, correlations can be drawn from co-occurring and semantically similar annotations across different evidences in the log. This can again help connect the dots between events, people, organizations, offenses and artifacts involved in a case. In addition, annotations can also be leveraged along with other analysis dimensions such as time and space. For example, the combination of annotations related to event types (e.g., *bank transaction*) can be analyzed from a time perspective to help re-construct how the sequence of events shaped a given case. This can be elaborated even further by considering the space perspective where event-related annotations are correlated to geographical locations plotted on a map, e.g., to help investigators precisely characterize the locations where the events took place. In fact, later in Section 3.3, we illustrate examples of how some of these interesting insights can be visualized using our implemented tool.

The tasks and examples discussed above provide an overview of how evidences are analyzed and annotated manually in an investigation. In this paper, we replace such manual event type annotations with *dynamic tags*, and we show how they can be *automatically recognized* from evidence items. We also showcase how our tool, *Case Walls*, can help facilitate insights and discovery in investigation processes. We discuss all this next.

3 Our Approach to Dynamic Event Type Recognition and Tagging

The evidence collection and analysis scenario discussed in the previous section show how tightly data can be connected to a case. Moreover, much of such data is nowadays being increasingly generated through digital means. At the same time, social and other Web-oriented platforms are promoting the creation of tremendous amounts of data on a daily basis. As discussed previously, a significant part of the investigation work, therefore, focuses on the analysis of data for the recognition of events relevant to a case, discovery of insights and presentation of such results to end-users (investigators) to support critical decision making within cases [36]. In this section, we propose a framework to support the process of automatically recognizing and tagging event-types from evidences collected as part of an investigation process.

The key rationale behind our proposed architecture is that much of the evidences in an investigation are described and represented through textual reports (e.g., phone call transcripts and e-mails). In our proposed solution, we thus focus on the processing and analysis of textual data that can provide insights into the case. As we will discuss next, we therefore leverage on NLP techniques [29][31] that can help us process such data for the automatic recognition and tagging of event-types.

Figure 2 illustrates our overall architecture, which consists of three main layers. Starting with ***Training Data Generation***, this layer mines existing domain-knowledge, relevant corpora and general purpose knowledge graphs to formulate seed data. People (domain experts or crowd) involvement in this layer ranges from filtering the extracted data to taking upon an active role in the processing of data in order to formulate a seed dataset that will help our automatic event recognition approach. Next, at the heart of the framework, the ***Data Processsing*** layer transforms raw data from evidence logs into semantically rich event-type representations. This layer leverages on state of the art NLP techniques [29][31] in order to support the automatic recognition of event-types out of the items found in the evidence log of a case. More specifically, we leverage on word embedding techniques [31] in order to encode various event-types of interest into a sematically rich Vector Space Model (VSM), similar to how words are represented as vectors through word embedding techniques.

Finally, at the ***Insights and Discovery*** layer, raw data is presented appropriately for end-users to understand and make decisions. A key component of this layer is the use of a *dynamic tagging* mechanism. Tagging in general is useful to assign semantic to data, which thereby enables powerful analysis (the simplest example is, if two items share the same tag, we can infer a relationship between them) [44]. Our architecture, however, transcends the idea of basic tagging and also introduces the notion of dynamic tags: These are tags that are automatically created from raw evidence logs. Another important component for insights and discovery is the use of visualization techniques. Accordingly, we propose supporting this by providing suitable visualization for associating the dynamic tags with evidence items in the log.

In addition to the above, an important feature common across the whole framework is ***Feedback and Learning***. In our proposal, this can be achieved in three ways: Firstly, the process for seed generation is not one-off but re-invoked to improve the system's perception of knowledge. The data being processed can be periodically *monitored* and *analyzed*, e.g., by domain experts, and *updated* accordingly based on their

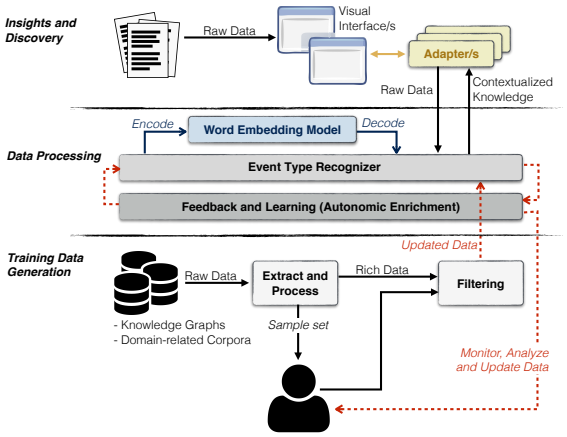

Fig. 2 Our framework for dynamic recognition and tagging of event-types for insights and discovery

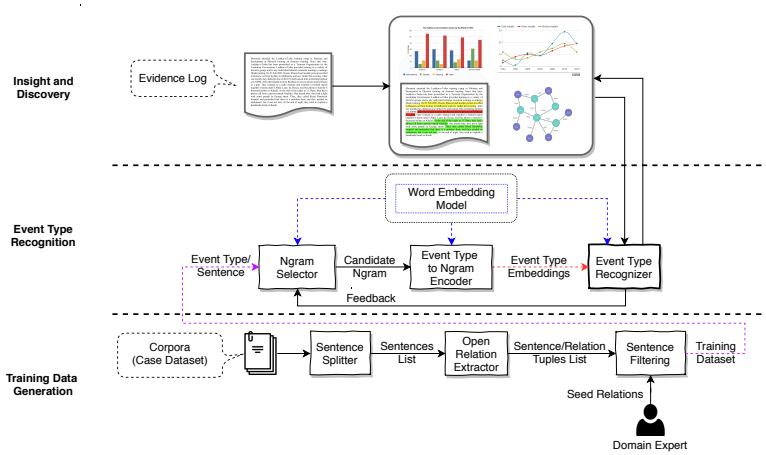

Fig. 3 Detailed architecture for Training Data Generation, Event Type Recognition, and Insight and Discovery.

feedback. Secondly, feedback can also be obtained from real-time data. For example, when investigation data is inputted for processing, the existing VSM can be updated to reflect, for instance, new event-types of relevance for a case. Thirdly, the use of dynamic tags also enable end-users (investigators) to accept or reject tags (e.g., if tags were added incorrectly), thus providing additional feedback to the system.

In the following sections, we present the details of our architecture starting from the Training Data Generation layer. We then move on to discuss the remaining layers for Event Type Recognition, and Insights and Discovery. We use Figure 3 as reference to explain in more details the components of each layer of our architecture.

3.1 Training Data Generation

Given the textual nature of much of the evidence items found in investigations, in this section, we describe the corpus (dataset) used to extract raw sentences and generate labeled sentences with event-types. Such sentences are used as training data for the event recognition technique we propose in the next section.

3.1.1 Datasets

The dataset we use for training purposes was obtained from past court cases (including both criminal and civil cases) in Australia, which are made publicly available by the Australasian Legal Information Institute (AustLII) [16]. While such court dataset does not contain police investigation data, it contains rich descriptions about events, facts, transcripts of communications and other relevant details of the case, which are similar to the content found in police investigation documents.

More specifically, in this work we use the pre-processed dataset available at the UCI Machine Learning repository [11]. This dataset contains a total of approximately 4K cases, where each case record is made available as an XML file. The XML file consists of a list of sentence nodes (e.g., *<sentence>In all these circumstances, subject to the applicants... </sentence>*) that contain text recorded in each case. Figure 4 shows an excerpt of one of these files. The file contains additional metadata, such as catchphrases

```
<case>
  ...
  <sentence id="s7">
    Its businesses are focused on media ...
  </sentence>
  <sentence id="s8">
    According to an independent expert's report in evidence ...
  </sentence>
  ...
</case>
```

Fig. 4 Excerpt of one of the case files in our case dataset.

```
[
  ...
  {
    "sentence": "While in the house they telephoned him and asked for his
                 whereabouts.",
    "eventType": "Phone_Call"
  },
  {
    "sentence": "The appellant arrived in [...] on 7 December 2003 ... ",
     " eventType ": "Travel_Movement"
  }
  ...
]
```

Fig. 5 Excerpt of the gold standard data showing two sentences and their corresponding event-types.

(summaries of the case) and links (URLs) to the original source (AustLII's website [16]). We ignore these metadata since they do not contain relevant sentences to support event-type recognition.

3.1.2 Training Data for Event Type Recognition

Armed with the dataset described above, we transform and process it into valid training data. For illustration purposes, consider an excerpt from our dataset below:

> *"Mr. Martens said that during 1987 whilst working part-time in Belmont [...]. However, in January 2002, Mr. Martens telephoned him at the offices of Mr. Hoffman, and engaged him as a tax accountant for him and his wife Victoria."*

In the paragraph above, the sentence *"...in January 2002, Mr. Martens telephoned him at the offices of Mr. Hoffman, and engaged him as a tax accountant..."* makes reference to an event of type *Phone Call*. In this layer of our framework, the goal is to mine our case dataset to identify sentences containing event-types and generate labeled data that can be used for training purposes.

First, we split the dataset using the *Sentence Splitter* (see the bottom layer in Figure 3). To do this, we use a Python library called splitter[1] (version 1.4). The output of this component consists in a list of sentences similar to the one exemplified above. Notice, however, that not all sentences will necessarily contain relevant event-types. Therefore, in the next step, the *Open Relation Extractor* uses relation extraction techniques [3] to help identify a list of sentences containing event-types (e.g., *Phone*

[1] https://github.com/berkmancenter/mediacloud-sentence-splitter

Call). To perform this task, we use OpenIE[2] (version 3.6.0), a component of the Stanford Core NLP Toolkit [29]. Relation extraction is the task of extracting tuples of the form <*subject, relation, object*> in which *subject* and *object* are related to each other by *relation*. The output of this component is a list of pairs $\{< sent_i, rels_j >_k\}$, where $sent_i$ represents a sentence (e.g., *"However, in January 2002, Mr. Martens telephoned him as ..."*) and $rels_j$ is a list of relation tuples identified in sentence *sent*. As an example, the *Open Relation Extractor* will produce a tuple <*Mr. Martens, telephoned, him*> for our exemplary sentence. The relations in these tuples (e.g., *telephoned*) can indicate the presence (or not) of an event-type of relevance (in this case, the verb *telephoned* indicates the presence of an event of type *phone call*).

The *Open Relation Extractor* extracts all sorts of relations from sentences, regardless of whether they refer to event-types of interest or not. In the sentence above, besides the relation tuple <*Mr. Martens, telephoned, him*>, additional relation tuples that are not related to our list of relevant event-types can also be extracted, such as <*Mr. Martens, engaged, him*>. The *Sentence Filtering* component (see Figure 3) helps in filtering out sentences that do not contain relevant events. In order to do this, it takes in input the list of sentences from the case dataset and the identified relations within each of them (i.e., $\{< sent_i, rels_j >\}$), as well as a list of curated seed relations (e.g., generated by domain-experts). It then keeps only sentences containing relations (e.g. *telephoned* or *contacted with*) found among the list of seed relations. The final list of remaining sentences is then labeled with the event-types they contain and passed onto the upper layer.

Following this approach, a total of *500* sentences have been extracted from our dataset, each one labeled with one of the event-types of interest (*Phone Call, Bank Transaction and Travel Movement*). In the following section, we discuss how this training data is exploited for the purpose of event-type recognition.

3.2 Event Type Recognition

In order to recognize event-types from evidence items, we first need to be able to find a suitable representation for event-types that are relevant in the context of investigations. The key intuition behind our proposed approach for event-type recognition is that, similar to how word embeddings [31] are used to represent words in a VSM, we can also encode event-types into vectors. The resulting VSM for event-types can be thus leveraged in order to help recognize event-types within investigation data. Using the architecture shown in Figure 3 as reference, we elaborate next on the details of how we achieve this.

3.2.1 Event-Type Vector Encoding

While general-purpose word embedding models often consider "words" as vectors, we have extended this idea to represent event-types in a VSM. We start by expressing event-types as a list of n-grams [30]. For example, n-grams like {*call on*}, {*ring back*}, {*phone call*}, {*later call*} *and* {*make contact on*} convey the idea that we are referring to a *Phone Call* event-type.

In order to build a vector for the event-type *Phone Call*, we therefore start by first building vectors for each of these n-grams. These individual vectors are then combined to formulate the event-type vector. Figure 6 shows the steps taken to build a vector for the *Phone Call* event-type. This is based on an initial set of seed n-grams provided by a human expert (e.g. {*call on*}, {*ring back*}, {*phone call*}, {*later call*},

[2] https://nlp.stanford.edu/software/openie.html

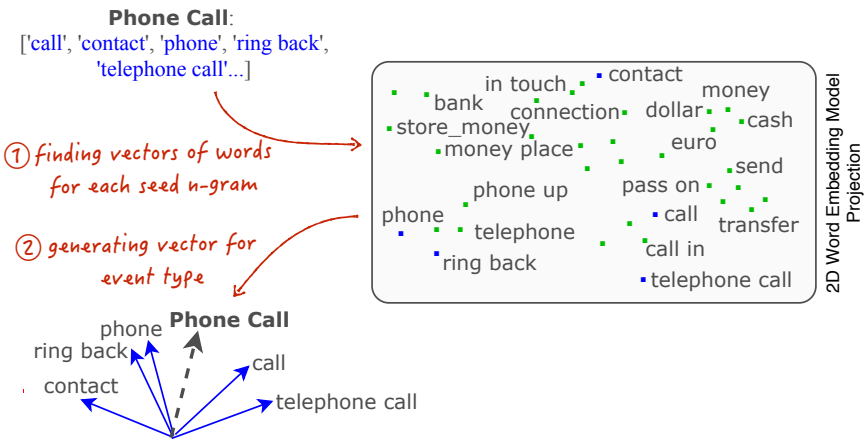

Fig. 6 Event Type Vector Encoding Using Seed n-grams

{*make contact on*}). Alternatively, this could also be sourced from an initial training data sample (e.g., see how the *Ngram Selector* component in Figure 3 is fed by the *Training Data Generation* layer). Using these seed n-grams, we then build vectors for each one of them by using a word embedding model [31] (see the *Event Type to Ngram Encoder* component in Figure 3).

Equation 1 formally describes the computations used to encode both individual n-grams and event-types as vectors:

$$v(ngram) = \frac{1}{n}\left(\sum_{k=1}^{n} v(w_k)\right) \; v(evtype) = \frac{1}{m}\left(\sum_{i=1}^{m} v(ngram_i)\right) \tag{1}$$

Here, $v(ngram)$ is a vector of an n-gram obtained by averaging the vectors (v) of each word in the ngram (w_k), and n is number of words in that n-gram. Then, $v(evtype)$ is a vector for an event-type generated by averaging a set of n-grams vectors (i.e., $v(ngram)$) corresponding to an event-type, and m is the number of n-grams linked to the event-type. The result of this is the encoding of event-types into a VSM. Next, we will discuss how we can fine-tune these event-type vectors in order to obtain a more precise representation of event-types.

3.2.2 Tuning Event-Type Vectors

In order to fine-tune our event-type vectors, we need to ensure that the underlying n-grams accurately represent each event-type. One way of achieving this is to build our n-gram vectors using sample sentences (and thus, n-grams) from our target domain. The intuition behind this is that the use of domain-specific corpora will help us obtain n-grams that are tightly connected to the terminology used within the domain.

For our Law Enforcement scenario, we therefore utilize the dataset of court cases introduced previously. In Section 3.1, we explained how we acquired this training dataset in more details. Our training dataset is a collection of sentences (e.g., *S.1*) that are labeled with at least one event-type.

> *"33 After receiving this email, Mr. Kaplan spoke to Mr. Gibson by telephone on the same day." (S.1)*

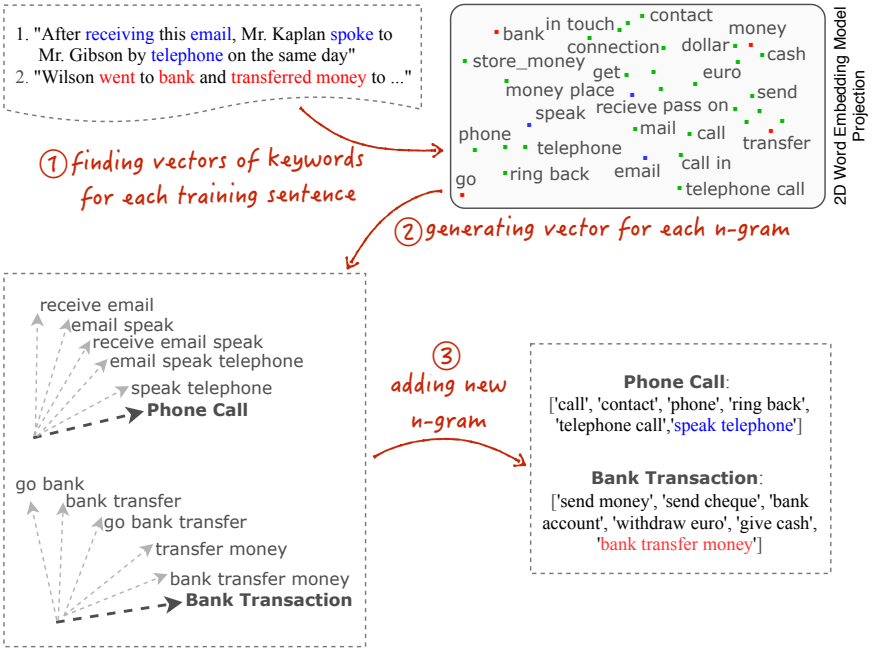

Fig. 7 Tuning Event Type Vector by using training dataset

Following our architecture in Figure 3, the *N-gram Selector* component extracts all *keywords* (*nouns* and *verbs*) from sentences of the corpora (see step 1 in Figure 7). To do this, it uses Stanford Part-Of-Speech (POS) Tagger[3] (version 3.9.2). Next, it removes stop words, and uses Spacy lemmatizer[4] (version 2.2) to get the root form of each remaining keyword. It then creates a list of n-grams out of these keywords. If we consider, e.g., sentence *S.1* above, possible n-grams are {*receive email, email speak, speak telephone, receive email speak, email speak telephone*}.

The n-grams are then converted into vector representations, which are then compared with the vector of existing event-types (step 2 in Figure 7). Next, the *N-gram Selector* component tries to find and select the top n-gram – the one that (i) has the closest vector to an event-type vector in our VSM, and (ii) has a similarity score greater than a given threshold[5]. After finding the top matching n-grams, the *N-gram Selector* component sends these n-grams to the *N-gram Encoder* (step 3) to update the corresponding event-type vector.

Learning from Real-time Data. The tunning of event-type vectors, as described above, is done using pre-existing training data. In addition to this, our proposed solution also allows for fine-tunning event-type vectors through continuous feedback from users and real-time investigation data. Here, as corpora from inputted evidence log is processed, feecback is sent to our *Event Type Recognizer* to update the existing event-type vectors. For example, the sentence *"Later, he, Ari, used his phone to transfer 6*

[3] https://nlp.stanford.edu/software/tagger.shtml
[4] https://spacy.io/api/lemmatizer
[5] We set an initial threshold to 70%; this parameter can be tuned as needed.

thousands dollars to Philip" contains the n-gram {*transfer, thousands, dollar*}, which can be considered relevant the event-type *Bank Transaction*. Such n-grams are sent to the *N-gram Selector* component to start the processing pipeline for updating the corresponding event-type vectors.

3.2.3 Event-Type Recognizer

The task of the *Event Type Recognizer* component (see Figure 3) is to detect possible event-type(s) that may appear in each sentence of an evidence log. This task starts by breaking down each paragraph of an evidence log into separate sentences[6] (step 1 in Figure 8). From each sentence, all keywords (nouns and verbs) are extracted. We specifically consider *verbs* because they refer to the performance of activities and actions (e.g., talking, travelling, transferring), and *nouns* (e.g., phone, bar, money) because they convey the general purpose of a sentence when considered in conjunction with verbs (e.g., talking on the phone, going to the bar, transferring money) [33][13]. Other elements such as person or date are also detected and tagged separately.

The extracted keywords are then converted into their root forms[7] and then used to build a list of possible n-grams[8] (e.g., {*phone conversation, conversation hospital, hospital pretend, pretend problem, phone conversation hospital, ...*}). The n-grams are then converted into vector representations (step 2 in Figure 8), which are then compared separately with our previously built event-types vectors (Sections 3.2.1 and 3.2.2). If the similarity between vectors (i.e., sentences' n-gram vector and event-type vectors) exceeds a specified threshold (α = %55 as an initial setup), the *Event Type Recognizer* records the corresponding event-types as *detected* event-types (e.g., *Phone Call* in step 3 of Figure 8). Notice that, if a sentence contains more than one event-type, the *Event Type Recognizer* will still be able to capture all of them (depending on the similarity threshold used). We also provide the possibility to flag event-types as *"suggested"*: This corresponds to event-types that score a similarity value that falls between β and α (i.e., for our initial setup, between $\beta = 40\%$ and $\alpha = 70\%$). These parameters can be fine-tuned to reflect the investigator's preferences in terms of precision and recall.

3.2.4 Event Information Extraction

In addition to recognizing events types, we also consider the possibility of extracting event information that can help provide a more holistic view on the events being analyzed. While different event-types can be characterized by different bits of information (e.g., a *bank transaction* may be characterized by the amount of money transferred, while a *phone call* may do so by the phone number making the call), we specifically focus on extracting three key entities that are relevant for any event during an investigation [12], namely, the *date/time* when the event happened, *location* where it took place, and the *parties* (e.g., people) involved in the event. To do so, we leverage on Information Extraction techniques [29] that allow us to recognize such entities from the textual descriptions of an event. More specifically, we leverage on named-entity recognition techniques [32] to perform this task and we rely on Stanford Named Entity Recognizer (NER) library[9] (version 3.9.2). For instance, take the sentence below, which contains an event of type *phone call*.

[6] see earlier Footnote 1

[7] see earlier Footnote 4

[8] We choose bigrams and trigrams, as based on our observation, averaging the vector of more than 3 words together results in an embedding that is not semantically meaningful

[9] https://nlp.stanford.edu/software/CRF-NER.html

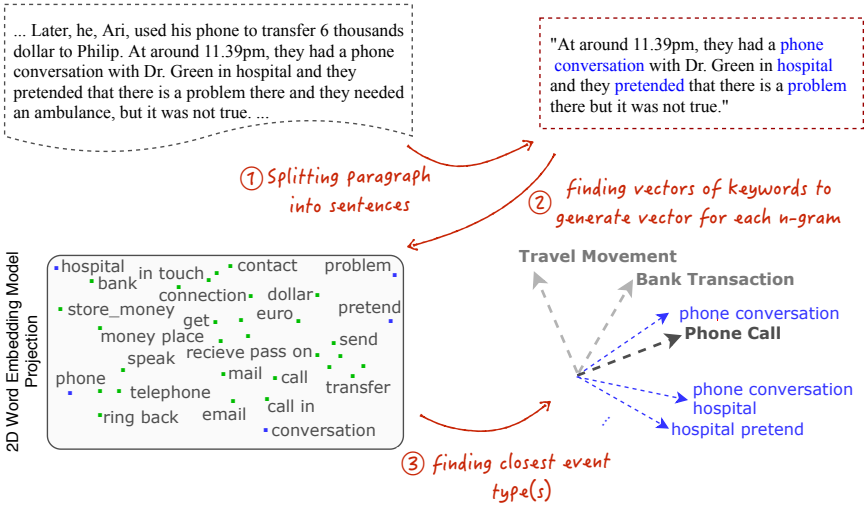

Fig. 8 Recognition of Event Types from Evidence Logs

> *"However, in January 2002, Mr. Martens telephoned Mr. Hoffman at his office in Lancaster, and engaged him as a tax accountant..."*

By performing named-entity recognition on the sentence above, we can extract the named-entities "*January 2002*" (date), "*Lancaster*" (location), and "*Mr. Martens*" and "*Mr. Hoffman*" (parties involved). The extracted named-entities can be used to annotate the recognized event, thus producing a richer representation thereof. As we will discuss in Section 3.3, these pieces of information are key to providing insights and guiding the discovery process in the investigation of cases.

3.3 Insights and Discovery

After event recognition and enrichment (with event-related information such timestamps and people involved), we can now illustrate how this information can be presented to end-users in the form of dynamic tags in *Case Walls*, our tool for supporting investigations in Law Enforcement (see the top layer in Figure 3). As mentioned earlier, tags inject semantics into data, and we utilize them to produce powerful visualizations for insights and discovery. Figure 9 shows our Case Walls in action. The **Digital Assistants Wall** (see the upper part of the figure) helps users to automatically recognize event-types within evidence items (see the text file in the **Workspace Wall**). Three event-types are automatically recognized by the Digital Assistant, namely, phone calls, travel movements and bank transactions. Furthermore, the corresponding statements are dynamically tagged in the text file displayed in the Workspace Wall (see the highlighted sentences). With the help of the Digital Assistants, the user can approve or reject the detected event-types, thus, proving feedback to the system to support learning.

From the dynamically recognized and tagged events, users can also derive useful insights that can help correlate events, people, locations and objects found in evidence items. At the bottom part of Figure 9 (see the **Visualization Widgets**), we show how **travel movements** are plotted on a timeline, along

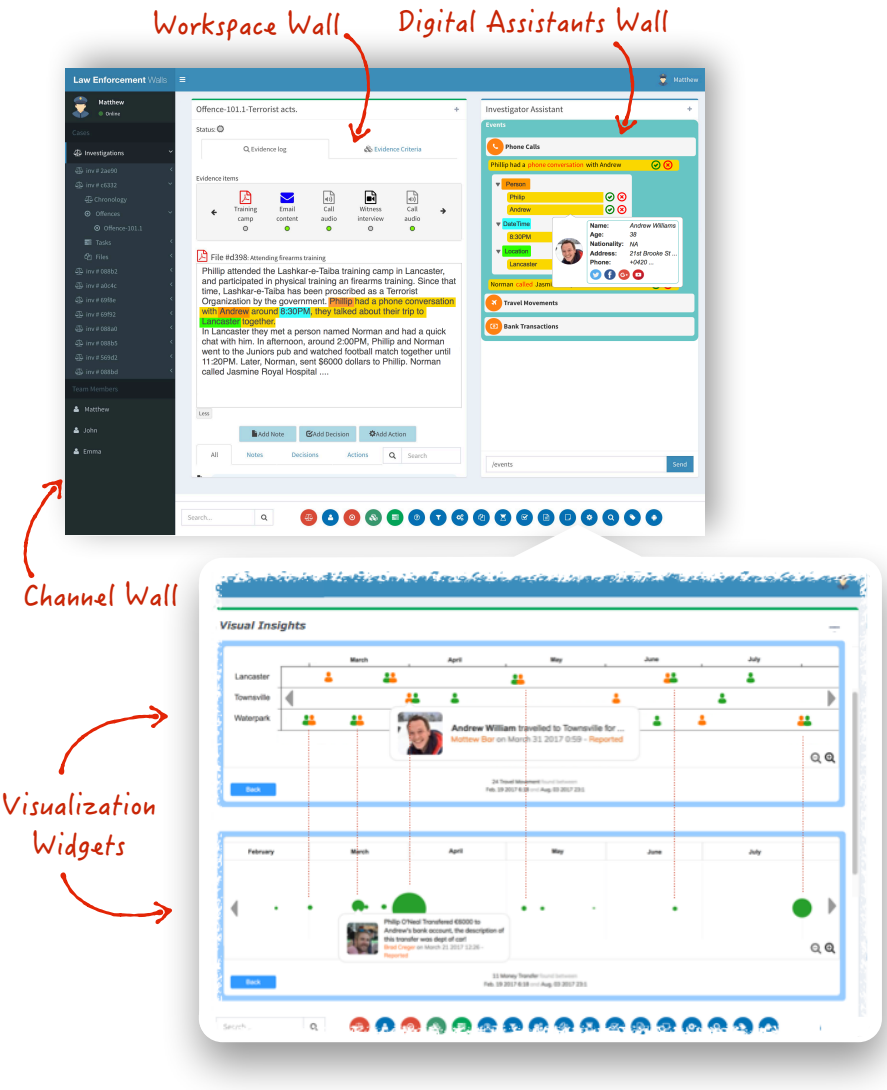

Fig. 9 Case Walls for Law-Enforcement

with other extracted elements such as **Person**, **location**, and **timestamps**. These elements correspond to the complementary information that is automatically extracted along with the recognized event-types, as discussed in the previous section.

The first timeline (top) presented in the Visualization Widgets shows a plot where the y-axis is set to **location**, while the x-axis to time. In this timeline, we are thus able to track the movement of **Persons**,

across time (as the x-axis). In the second timeline (bottom), we now plot **bank transaction** event-types. We set the y-axis using a formula for the frequency of these event-types (which determines the size of the green dots), again across time (as the x-axis). By juxtaposing these two timelines side-by-side we are thus able to *discover* that *after each travel movement and meeting between Andrew and Philip, there is a bank transaction shortly thereafter.*

The example above is just one instance of the type of insights investigators can gain by using our framework. Similar widgets can also be developed to support insights and discovery, e.g., to further connect events with offences, allegations and even past (or current) investigations. In the next section, we turn our attention to experiments that help validate the effectiveness of the techniques proposed for the automatic recognition of events.

4 Experiments

We demonstrate the effectiveness of our approach by conducting three experiments to test the precision of the proposed event-type recognition approach. We provide the details of our experiments next.

4.1 Event-Type Recognition

The aim of this experiment is to evaluate the ability of the *Event Type Recognizer* to identify events in our gold standard dataset consisting of *500* sentences (Figure 5), each labeled with one of *three* event-types: *Phone Call, Travel Movement and Bank Transaction*. The gold standard dataset is generated from our 4K cases dataset (see Section 3.1.2) by combining relation extraction techniques [3] and manual curation in order to guarantee that sentences are correctly tagged with the appropriate event types.

4.1.1 Conventional Validation

We evaluate the performance of the *Event Type Recognizer* through a conventional validation approach [46], where we split the gold standard dataset into 50% training and 50% test sets. We measure the standard Precision, Recall and F-Score by comparing both the output sentences and the corresponding event-types recognized to the gold standard dataset. Furthermore, we explore the performance of the approach as we increase the relative size of the training set (w.r.t. the test set) from 0% to 50%. The aim of this experiment is to show the effect of training on the performance while the *Event Type Recognizer* is being used. Notice that the thresholds to identify an event-type as *detected* is $\alpha = \%55$, and as *suspected* is $\beta = \%40$. The evaluation results are shown in Figure 10. Here, the precision for all three event-types is reasonably high even with 0% training set, i.e., by only using vectors built based on the initial set of seed n-grams (section 3.2.1). The precision amounts to *0.93, 0.80* and *0.90* for *Phone Call, Travel Movement* and *Bank Transaction*, respectively. As we increase the percentage of training set (from 0% to 50%), we can see that precision improves to 0.99, 0.87 and 1.00 for *Phone Call*, *Travel Movement* and *Bank Transaction*, respectively.

Recall, prior to training, reaches the value of *0.50* for *Bank Transaction*. Instead, for *Phone Call* and *Travel Movement* the values are 0.91 and 0.88, respectively. This low recall is caused by the fact that we have a rather small set of n-grams to start with for *Bank Transactions* in comparison to the the other two event-types (*Phone Call* and *Travel Movement* events where found at a higher frequency in our training dataset). However, as the system learns a higher number of n-grams, the recall goes up to *1.00* (at a 50% training set size).

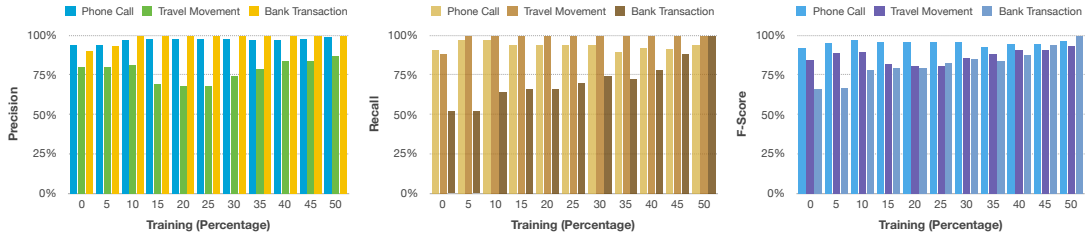

Fig. 10 Event Type Recognizer - Precision/Recall/F-Score for 0-50% training set (testing set is fixed on 50% of gold dataset)

4.1.2 K-Fold Cross Validation

In this experiment, we evaluate the performance of the *Event Type Recognizer* using k-fold cross validation [46], where we randomly partition the gold standard dataset into k equal-sized subsets. We choose $k = 5$ and perform 5 rounds (folds) of training and testing. In each round, we choose one different subset at a time as test set, while we train our vectors on the remaining 4 subsets. For each fold, the results produced on the test set are used to compute precision, recall and F-score (using our golden data as reference), which are then averaged to estimate the overall performance of the *Event Type Recognizer*. For this evaluation we use an experimental setting of 80% training, 20% testing. The results of our cross validation test are reported in Table 2. The average performance show very promising, where we have *Phone Call (0.94)*, *Travel Movement (0.91)* and, interestingly, *Bank Transaction (1.00)*. The recall, on the other hand, is also optimistic; it is above 0.95 for both *Phone Call* and *Travel Movement*, while for *Bank Transaction* recall reaches 0.88. These results show that if we train the system to learn more precise n-grams, it can indeed boost performance.

4.1.3 Human Validation

To further evaluate the accuracy of our approach, we ran the system on the 4K cases dataset, and then randomly picked a total of 500 event-type recognitions. Then, we asked two participants (Postdoctoral researchers) to independently classify each recognized event-type as *correct* or *incorrect*. To measure the accuracy of event recognizer, we used Cohen's Kappa coefficient (κ) [9], which measures the agreement between two individual raters. Typical interpretation of such coefficient in relation to the agreement levels are *Poor* ($\kappa < 0.20$), *Fair* ($0.20 \leq \kappa < 0.40$), *Moderate* ($0.40 \leq \kappa < 0.60$), *Good* ($0.60 \leq \kappa < 0.80$), *Very good* ($0.80 \leq \kappa \leq 1.00$) [2]. Kappa compares an *Observed Accuracy* with an *Expected Accuracy (random chance)* in a confusion matrix. As it considers *random chance*, it is less misleading than simply using accuracy as a metric. The results shows that both participants agreed on 96.05% of all the recognitions ($\kappa = 0.77$). The overall average accuracy was 88.4% for *Phone Call*, 80.2% for *Travel Movement* and 77.1% for *Bank Transaction*.

4.1.4 Effect of Word Embedding Model

The aim of this experiment is to see the impact of using different word embedding models (trained on different corpora) on the performance of our *Event Type Recognizer*. We first shuffled the gold standard dataset and split it into two sets: 70% training, 30% test. We then evaluated the performance of *Event*

Table 2 Event Type Recognizer - Precision(P)/ Recall(R)/ F-Score(F)/ Average(Avg) for 5-Fold Cross-Validation

	Phone Call			***Travel Movement***			***Bank Transaction***		
Folds	P	R	F	P	R	F	P	R	F
F1	0.89	1.00	0.94	0.90	0.98	0.94	1.00	0.83	0.91
F2	1.00	0.90	0.95	0.89	1.00	0.94	1.00	0.63	0.77
F3	0.94	0.97	0.96	0.94	1.00	0.97	1.00	0.83	0.91
F4	0.97	0.88	0.92	0.84	1.00	0.91	1.00	1.00	1.00
F5	0.89	1.00	0.94	0.96	1.00	0.98	1.00	1.00	1.00
Avg	**0.94**	**0.95**	**0.94**	**0.91**	**1.00**	**0.95**	**1.00**	**0.86**	**0.92**

Type Recognizer - in terms of Precision and Recall - while using different word embedding models. We present below the list of embedding models used for this test:

- GoogleNews: A publicly available word2vec model trained on Google News articles[10] with about 3 million words and phrases.
- WikiNewsFast[6]: A previously fastText[11] trained model on Wikipedia (2017), UMBC webbase corpus and statmt.org news with 16 billion words.
- GigaGlove: A pre-trained model on Wikipedia (2014) and Gigaword (5th edition) by using Glove[12] algorithm.
- Numberbatch[39]: A pre-trained model on ConceptNet[13], an open multilingual knowledge graph.
- TF-IDF[14]: It is a baseline model to measure similarity of sentences (documents) by representing words according to their importance/frequency in corpus (training set).
- WikiW2V: Our own word embedding model trained on Wikipedia.

The results are reported in Table 3. Here, we can see that the *Event Type Recognizer* performs very well when it uses an embedding model trained on general purpose corpora (e.g., Wikipedia). Instead, the performance is dramatically reduced when traditional keyword matching techniques are used. For instance, for *Phone Call*, the *Event Type Recognizer* reaches its best Precision (0.96) and Recall (1.00) when it uses WikiW2V model. Likewise, it reaches to maximum Precision score in *Travel Movement* when WikiW2V is used. Furthermore, best Recall happens when either WikiNewsFast or Numberbatch model is used. In addition, in *Bank Transaction*, Precision faces its highest score while the pre-trained Glove embedding model is used. Finally, Recall has its best score in both WikiNewsFast and WikiW2V models.

While results are very positive when the *Event Type Recognizer* uses general purpose word embedding models, performance is rather modest when the TF-IDF embedding model is employed. The reason of such low performance is the *high percentage of False Positives*. As TF-IDF considers only (i) frequency of a word in each training sentence, and (ii) weight of rare words across all training sentences in the corpus, it does not exploit the *contextual meaning* of words in sentences. This leads to losing semantic similarities/relations between words and sentences. As a practical example of this, consider the sentence,

[10] https://code.google.com/archive/p/word2vec/
[11] https://github.com/facebookresearch/fastText
[12] https://nlp.stanford.edu/projects/glove/
[13] http://conceptnet.io/
[14] Term Frequency Inverse Document Frequency

Table 3 Event Type Recognizer - Precision(P) and Recall(R) while using different embedding models

	Phone Call		***Travel Movement***		***Bank Transaction***	
Models	P	R	P	R	P	R
GoogleNews	0.73	0.90	0.48	1.00	0.79	0.38
WikiNewsFast	0.92	0.94	0.84	**1.00**	0.95	**1.00**
GigaGlove	0.83	0.95	0.86	0.96	**1.00**	0.80
Numberbatch	0.81	0.95	0.55	**1.00**	0.72	0.84
WikiW2V	**0.96**	**1.00**	**0.93**	0.93	0.97	**1.00**
TFIDF	0.80	0.33	0.73	0.27	0.78	0.46

"Mr Green talked with his secretary in a meeting, and asked her to send his phone number to Mr Ari before 2PM". Here, although the sentence has four important keywords (*talked, asked, phone, number*) that are somewhat related to a phone communication, the event-type contained in the sentence cannot be recognized as a *"Phone Call"*.

5 Discussion

The framework discussed in this paper is part of a larger effort for supporting insights and discovery in police investigations. Such an initiative is realized through our prototype platform, *Case Walls for Law Enforcement* (or just *Case Walls* for short). Our platform provides a collaborative, assistive and analyst-friendly environment to both manage and analyze case data. We do this through features that can be categorized along the following dimensions:

- **Cognitive intelligence**. First, Case Walls provides a powerful set of computational techniques that provide support to cognitive tasks that are performed as part of police investigations. Such tasks typically require the ability to obtain insights into cases in order to make decisions. In this context, our platform can help in tasks such as the identification of relevant events in textual data (e.g., from police narratives). In addition, it is also able to automatically detect the presence of a potential offense (i.e., violation of a law) from such data, and more so correlate the elements of an offense (e.g., fraud) to a set of events (e.g., bank transaction to a certain person). Finally, it also provides support for natural language (NL) searches on top of insights obtained from such intelligent processing of data (e.g., search for events and people within a case).
- **End-user Digital Assistance.** Layered upon cognitive intelligence, Case Walls then provides a set of digital assistants (in the form of chatbots) that can help investigators perform their tasks. For example, by leveraging on the event and offense detection intelligence discussed before, the digital assistants can facilitate the tagging of evidences by automatically recommending tags that can be used for annotating evidences. Other types of digital assistance include the support for investigation tasks creation (e.g., habitation checks), entity search (e.g., searching for persons of interest) and NL queries (e.g., to search for offenses). We designed and implemented a range of digital assistance techniques including: (i) interactive tagging where investigators are automatically suggested tags, but can also add or remove to teach the system; (ii) Natural-language search capabilities (e.g. search evidence, events, offenses); (iii) context-awareness and producing required information as needed (e.g. suggesting evidence to substantiate an offense); and (iv) investigation briefing.

The dynamic event type recognition and tagging approach proposed in this paper plays an important role in the realization of the features above. While this paper discussed our approach in the context of event type recognition, the techniques can be easily extrapolated to other important investigation tasks such as tagging of evidence items with potential offences (as discussed above).

The experimental results presented in the previous section demonstrate the effectiveness of our approach for auto-recognition of event-types. The proposed technique, however, comes with its own *limitations*. As each event-type vector evolves over time (learning/adding new n-grams), it may shift away and no longer represent the event-type precisely. In order to keep vectors precise, we can leverage *Reinforcement Learning* [17] to learn from positive/negative feedbacks from investigators, to reward/punish the system by adding/removing n-grams.

Another issue that is worth noting is that the growing number of event types might degrade the recognition performance. Therefore, as the number of event types expands, recognition tasks will become more rigid and it may need a supervised approach. We aim to explore this concern as we collect more training data. We plan to also conduct larger scale experiments with training in the midst of real-life investigations. Finally, in order to apply our framework to other domains, further experiments and evaluations are necessary, e.g., using existing publicly available datasets (news archives, tweets, etc).

6 Related work

Data-driven insights in the legal domain. Several studies can be found where NLP and Machine Learning (ML) techniques are used for the analysis of legal corpora for organizational, understanding and prediction purposes. Works like [26,35] focus on *legal document clustering*, where, e.g., Lu and Conrad [26] propose to cluster legal documents using a classification-based approach that is equipped with topic segmentation. The work demonstrates that clustering legal documents can be done effectively and efficiently with this approach by leveraging on metadata currently available in legal corpora. On a different front, the problem of *document summarization* has also been addressed in the legal domain [34,14,21]. Here, Posley et al. [34] propose CaseSummarizer, a system that helps in the automatic summarization of legal texts. In their approach, the authors propose to leverage on existing, well-established methods from natural language processing (e.g., part-of-speech tagging and TF-IDF [30]) and domain-specific constructs and show that CaseSummarizer perform better that non-domain specific alternatives.

The problem of *outcome predictions for court cases* has also caught the attention of researchers [41, 27]. Here, Luo et al. [27] focus on predicting charges in criminal cases taking into account both the facts described in cases and the articles stemming from the criminal law. They propose an attention-based neural network that help predict both the charges and the relevant articles applicable to a case. Finally, *action, event and offense identification* has also been explored in the legal domain, however, within a broader scopus. For example, on the action and event identification front, Soria et al. [38] propose SALEM, a framework for automatically annotating legal text with semantics. The framework allows for the annotation of entities and actions, and the connection of these elements to specific types of regulations (e.g., obligations, penalties and prohibitions) as emerged from legal texts. Similarly, Liu and Liao [23] address a similar problem in the context of civil law. They use instance-based classification techniques on top of a Chinese legal corpora to classify documents based on the law articles involved in the case.

The works discussed above focus mostly on analyzing collections of legal documents and providing a holistic and retrospective insights into them, except for the line of work on predicting outcomes of court cases [41,27]. Our work instead focuses on utilizing existing corpora for training event-type embeddings

for recognition, with the final aim of providing insights into cases and proactively assisting investigators as they proceed with their investigative tasks.

Data-driven insights in law enforcement. The law enforcement domain has also traditionally benefited from NLP and ML for supporting criminal investigation tasks. In the area of *crime and authorship*, Rahman et al. [5] address the problem of topic and author identification as well as the level of contribution of authors to topics over time on a dataset made of chat logs. Two extensions are proposed to standard, probabilistic topic modeling [45]. Namely, the LDA-topics over time (LDA-TOT) and the Author-Topics over time (A-TOT). Zhen et al. [47], instead, investigate authorship in the context of cybercrime by analyzing the content of e-mails and forums. They study the use of traditional ML algorithms (C4.5, Neural Networks and SVM) [46] with different combinations of features, including style markers (e.g., number of uppercase characters), structural features (e.g., special characters) and content-specific features (e.g., references to prices in subjects of e-mails). The combination of structural features and style markers seems to produce the best results.

Other works focus on *spatio-temporal, textual analysis* [25,15]. For example, Helbich et al. [15] study the use of narrative crime reports to build maps where such documents are clustered and correlated to a geographical space in order to get insights into the geography of crime. The authors used a combination of self-organizing maps [19] and point pattern analysis [37] the help identify clusters of documents and map them into geographical locations. Similarly, Liu et. al [25], propose a search engine that leverages on spatio-temporal visualizations and information retrieval to help investigators query and geographically render crime reports, where techniques from information extraction [30], indexing [43] and clustering [46] are employed.

Works like [42,8,20], instead, explore the use of *entity and relation extraction techniques to help crime investigation tasks*. Here, Sun et al. [42] propose to leverage on information extraction techniques [30] to extract terrorism-related events from documents, while Chau et al. [8] propose to extract relevant entities found in police reports through lexical rule patterns obtained with a neural network trained for the purpose. In the same line, Ku et al. [20] propose to extract crime investigation information such as locations, people and vehicles using standard NLP techniques [30]. Finally, *crime matching and clustering* has also been addressed in the context of criminal investigations. For example, works like [18,10] both leverage on text clustering techniques for crime matching and forensic analysis. The former addresses the problem of matching crimes and criminals to previous cases, while the latter focuses on demonstrating the usefulness of clustering techniques to support information retrieval and authorship.

In our work, we utilize recent advances in NLP and ML techniques, including the combination of powerful techniques such as Word2vec [31]. The framework we propose, therefore, not only leverages on the knowledge that can be obtained from law enforcement corpora alone but can also benefit from the more general knowledge (i.e., general word embeddings and knowledge graphs). Together this equips our framework with robust tools that can help gain useful insights into cases.

7 Concluding Remarks

The approach discussed in this paper represents a step toward the practical realization of *Cognitive Case Management (CCM)* for Law Enforcement, where data, NLP, AI and computational processing power are consider first-class citizens. The implications of leveraging on such technologies in this context can be translated into an increased productivity and efficiency, and improved decision making and insights when managing cases.

The *increased productivity and efficiency* comes from the help that the technologies above can provide in facilitating the automation of repetitive, tedious and lengthy tasks. Without such help, investigators not only risk the waste of a large amount of time that could be otherwise spent in tasks of higher cognitive nature (e.g., identifying patterns in a case) but also getting exposed to the risk of missing important bits of information that could be relevant to the case. This is particularly relevant in cases involving a very large corpora. Bringing these functionalities into the context and operations of investigators (e.g., identifying events while entering allegations into the system) can help avoid switching contexts and create a deeper awareness and insights into the current task.

The *improved decision making and insights* can be obtained by leveraging on the outcomes of the cognitive assistance provided by the discussed framework. For example, further data analytics techniques can be applied on the recognized events in order to identify named entities (e.g., organizations and locations) and relations among them, tags and summaries can be automatically derived from event descriptions, and patterns can be recognized from event dynamics. Finally, deeper exploration of cases can be supported on top of the insights above in order to accelerate discovery, pattern recognition and linking.

The future has in store many further exciting opportunities. Such as codifying offenses from legislation, and methods for correlating collection of events and mapping to offenses. This can help automate the task of determining whether the elements of an offense can indeed be substantiated. Furthermore, we are confident the foundations of our work can be applied and extended to many other domains involving investigative tasks (e.g., science and research).

Acknowledgement. We acknowledge Data to Decisions CRC (D2D-CRC) for funding this research.

References

1. Al Mutawa, N., Baggili, I., Marrington, A.: Forensic analysis of social networking applications on mobile devices. Digital Investigation **9**, S24–S33 (2012)
2. Altman, D.G.: Practical statistics for medical research. CRC press (1990)
3. Angeli, G., Premkumar, M.J.J., Manning, C.D.: Leveraging linguistic structure for open domain information extraction. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), vol. 1, pp. 344–354 (2015)
4. Baber, C., Smith, P., Cross, J., Hunter, J.E., McMaster, R.: Crime scene investigation as distributed cognition. Pragmatics & Cognition **14**(2), 357–385 (2006)
5. Basher, A.R.M., Fung, B.C.: Analyzing topics and authors in chat logs for crime investigation. Knowledge and information systems **39**(2), 351–381 (2014)
6. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606 (2016)
7. Bolukbasi, T., Chang, K., Zou, J.Y., Saligrama, V., Kalai, A.: Man is to computer programmer as woman is to homemaker? debiasing word embeddings. CoRR **abs/1607.06520** (2016). URL http://arxiv.org/abs/1607.06520
8. Chau, M., Xu, J.J., Chen, H.: Extracting meaningful entities from police narrative reports. In: Proceedings of the 2002 annual national conference on Digital government research, pp. 1–5. Digital Government Society of North America (2002)
9. Cohen, J.: A coefficient of agreement for nominal scales. Educational and psychological measurement **20**(1), 37–46 (1960)
10. Decherchi, S., Tacconi, S., Redi, J., Leoncini, A., Sangiacomo, F., Zunino, R.: Text clustering for digital forensics analysis. In: Computational Intelligence in Security for Information Systems, pp. 29–36. Springer (2009)
11. Dheeru, D., Karra Taniskidou, E.: UCI machine learning repository (2017). URL https://archive.ics.uci.edu/ml/datasets/Legal+Case+Reports. [Online; last accessed 26 March 2019]
12. Dobash, R.E., Dobash, R.P.: The nature and antecedents of violent events. The British Journal of Criminology **24**(3), 269–288 (1984)

13. Fast, E., McGrath, W., Rajpurkar, P., Bernstein, M.S.: Augur: Mining human behaviors from fiction to power interactive systems. In: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, pp. 237–247. ACM (2016)
14. Galgani, F., Compton, P., Hoffmann, A.: Citation based summarisation of legal texts. In: Pacific Rim International Conference on Artificial Intelligence, pp. 40–52. Springer (2012)
15. Helbich, M., Hagenauer, J., Leitner, M., Edwards, R.: Exploration of unstructured narrative crime reports: an unsupervised neural network and point pattern analysis approach. Cartography and Geographic Information Science **40**(4), 326–336 (2013)
16. Insititute, A.L.I.: Austlii: Free, comprehensive and independent access to australasian law. Available: www.austlii.edu.au (2018). [Online; accessed 7-May-2018]
17. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. Journal of artificial intelligence research **4** (1996)
18. Keyvanpour, M.R., Javideh, M., Ebrahimi, M.R.: Detecting and investigating crime by means of data mining: a general crime matching framework. Procedia Computer Science **3**, 872–880 (2011)
19. Kohonen, T.: The self-organizing map. Proceedings of the IEEE **78**(9) (1990)
20. Ku, C.H., Iriberri, A., Leroy, G.: Natural language processing and e-government: crime information extraction from heterogeneous data sources. In: Proceedings of the 2008 international conference on Digital government research, pp. 162–170. Digital Government Society of North America (2008)
21. Kumar, R., Raghuveer, K.: Legal document summarization using latent dirichlet allocation. International Journal of Computer Science and Telecommunications **3**, 114–117 (2012)
22. Lenci, A.: Distributional semantics in linguistic and cognitive research. Italian journal of linguistics **20**(1), 1–31 (2008)
23. Liu, C.L., Liao, T.M.: Classifying criminal charges in chinese for web-based legal services. In: Asia-Pacific Web Conference, pp. 64–75. Springer (2005)
24. Liu, H., Chen, S., Kubota, N.: Intelligent video systems and analytics: A survey. IEEE Transactions on Industrial Informatics **9**(3), 1222–1233 (2013)
25. Liu, X., Jian, C., Lu, C.T.: A spatio-temporal-textual crime search engine. In: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 528–529. ACM (2010)
26. Lu, Q., Conrad, J.G., Al-Kofahi, K., Keenan, W.: Legal document clustering with built-in topic segmentation. In: Proceedings of the 20th ACM international conference on Information and knowledge management. ACM (2011)
27. Luo, B., Feng, Y., Xu, J., Zhang, X., Zhao, D.: Learning to predict charges for criminal cases with legal basis. arXiv preprint arXiv:1707.09168 (2017)
28. Majone, G.: Evidence, argument, and persuasion in the policy process. Yale University Press (1989)
29. Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., McClosky, D.: The stanford corenlp natural language processing toolkit. In: Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations, pp. 55–60 (2014)
30. Manning, C.D., Schütze, H.: Foundations of statistical natural language processing. MIT press (1999)
31. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp. 3111–3119 (2013)
32. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Lingvisticae Investigationes **30**(1), 3–26 (2007)
33. Pandita, R., Xiao, X., Zhong, H., Xie, T., Oney, S., Paradkar, A.: Inferring method specifications from natural language api descriptions. In: Software Engineering (ICSE), 2012 34th International Conference on, pp. 815–825. IEEE (2012)
34. Polsley, S., Jhunjhunwala, P., Huang, R.: Casesummarizer: A system for automated summarization of legal texts. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations, pp. 258–262 (2016)
35. Raghuveer, K.: Legal documents clustering using latent dirichlet allocation. IAES Int. J. Artif. Intell **2**(1), 34–37 (2012)
36. Richard, T.: Qualitative versus quantitative methods: Understanding why qualitative methods are superior for criminology and criminal justice. Journal of Theoretical and Philosophical Criminology **1**(1) (2009)
37. Rowlingson, B.S., Diggle, P.J.: Splancs: spatial point pattern analysis code in s-plus. Computers & Geosciences **19**(5), 627–655 (1993)
38. Soria, C., Bartolini, R., Lenci, A., Montemagni, S., Pirrelli, V.: Automatic extraction of semantics in law documents. In: Proceedings of the V Legislative XML Workshop, pp. 253–266 (2007)
39. Speer, R., Chin, J., Havasi, C.: Conceptnet 5.5: An open multilingual graph of general knowledge. In: AAAI, pp. 4444–4451 (2017)
40. Stotland, E., Pendleton, M.: Workload, stress, and strain among police officers. Behavioral Medicine **15**(1), 5–17 (1989)
41. Sulea, O.M., Zampieri, M., Vela, M., van Genabith, J.: Predicting the law area and decisions of french supreme court cases. arXiv preprint arXiv:1708.01681 (2017)
42. Sun, Z., Lim, E.P., Chang, K., Ong, T.K., Gunaratna, R.K.: Event-driven document selection for terrorism information extraction. In: International Conference on Intelligence and Security Informatics, pp. 37–48. Springer (2005)

43. Tao, Y., Papadias, D.: Efficient historical r-trees. In: Scientific and Statistical Database Management, 2001. SSDBM 2001. Proceedings. Thirteenth International Conference on, pp. 223–232. IEEE (2001)
44. Vig, J., Sen, S., Riedl, J.: The tag genome: Encoding community knowledge to support novel interaction. ACM Transactions on Interactive Intelligent Systems (TiiS) **2**(3), 13 (2012)
45. Wallach, H.M.: Topic modeling: beyond bag-of-words. In: Proceedings of the 23rd international conference on Machine learning, pp. 977–984. ACM (2006)
46. Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann (2016)
47. Zheng, R., Qin, Y., Huang, Z., Chen, H.: Authorship analysis in cybercrime investigation. In: International Conference on Intelligence and Security Informatics, pp. 59–73. Springer (2003)