A.I. for Online Criminal Complaints: From Natural Dialogues to Structured Scenarios

Floris Bex, Joeri Peters and **Bas Testerink**[1]

Abstract. There exists a mismatch between the sort of crime reports that police would prefer to have and the stories people tell when filing a criminal complaint. Modern crimes such as trade fraud can be reported online, but a combination of static interfaces and a follow-up process that is dependent on manual analysis hamper the intake and investigation process. In this paper, we present our project Intelligence Amplification for Cybercrime (IAC), in which we aim to apply various AI techniques to allow natural dialogues about fraud cases. In this way, different parties such as citizens registering a complaint and police investigators can interface with cases composed of scenarios and evidence through natural language dialogues. This will not only solve an urgent practical problem, but also lead to new insights regarding computational models of evidence assessment.

1 Introduction

Reasoning in police investigations is a complex process, which consists of collecting, organizing and assessing a mass of unstructured and unreliable evidence and scenarios in a case [11]. Artificial Intelligence has proposed various scientifically founded ways of treating evidence using, for example, Bayesian networks [12, 22] or non-monotonic logics [7, 24, 15]. One problem for these A.I. models is that most people involved in the investigative process (e.g. detectives, prosecutors, witnesses) do not have the background to be able to to construct and utilize logical or probabilistic models of a case. Instead, the focus in real cases is often on more linguistically oriented concepts such as *arguments* and *scenarios*, often rendered informally (e.g. natural language) or semi-formally (e.g. mind-maps, argument maps). While recent research has tried to integrate arguments and scenarios with logic and probability theory [29, 23, 28], there still exists a clear gap between real investigations and more formal models [27]. This not only limits the practical applicability of A.I. models, but also makes it very difficult to validate whether formal models are useful and appropriate for investigative and decision-making practices (cf. [26, 25]). What is needed are technologies and theories for the process of investigation that bridge the gap between natural language interfaces and more formal models of evidential reasoning. This paper discusses such technologies and theories in light of a practical application, namely the improvement of online criminal complaints and the subsequent investigation.

Our project Intelligence Amplification for Cybercrime (IAC) aims to develop smart technologies to improve the online intake of criminal complaints and the subsequent investigations on the topic of e-crime and cybercrime. The possibility of reporting a crime online is relatively new in the Dutch police organisation, and it is currently only possible to report a few types of crime. One of these types concerns so-called e-crime, online fraud cases such as fake webshops and malicious second-hand traders. There are about 40,000 complaints about these types of cases every year, and while the damages in each individual case are usually quite small (around 50-100 euros), it pays to follow up on such complaints, particularly because suspects may be part of a larger criminal organization. The high volume and relatively low detail of such cases thus makes them ideal for online complaints and further automated processing.

In this paper, we sketch the outline of a system for reporting e-crime. In its current incarnation, this system consists of a dialogue interface and a module that translates structured and unstructured free text input from the dialogue interface to knowledge graphs, a labelled graph containing the entities, events and relations in a case. Citizens that want to file a complaint can thus tell a story about why they think they were victims of fraud. This story is then automatically translated to a graph that contains the evidence and (possible) scenarios in the case. This graph can then be used for further formal analysis, or to ask questions about the case in the dialogue, further eliciting relevant information from the person who makes the complaint.

The rest of this paper is structured as follows. In Section 2, we describe the application domain of online trade fraud by giving some examples and discussing the current intake process for criminal complaints. In Section 3, we explain the general architecture of our solution. Sections 3.3 and 3.2 delve into the applied computational linguistics and the application of argumentation dialogue literature. We outline in Section 3.4 how the improved structured data can support the police processes that are involved in online fraud. Finally, we conclude with our conclusions and future plans in Section 4.

2 Online Trade Fraud

The Dutch National Police has a number of possibilities for registering a criminal complaint online. Most of the crimes that can be reported online are low-profile, high volume crimes for which there is no clear suspect – for example, bike theft or petty vandalism. However, the National Service Centre E-Crime of the Dutch Police is currently involved in a pilot where citizens can file a complaint for online fraud cases, such as fake webshops and malicious second-hand traders, where there is a clear indication of a suspect (usually because the victim has transferred money to a bank account).

2.1 Typical Trade Fraud Scenarios

As examples of fraud scenarios, consider the types of fraud that take advantage of (first- and) second-hand auction websites, of which

[1] Utrecht University, the Netherlands, email: {F.J.Bex,J.G.T.Peters,B.J.G.Testerink}@uu.nl

ebay.com is probably the most recognised. A similar auction site that is very popular and well-known in the Netherlands is called Marktplaats(.nl) (lit. market place), which we will take as the example. The most obvious type of trading fraud is when swindler Y creates an ad on Marktplaats, advertising a product. Victim X responds to this ad and decides to buy the good. X then transfers the agreed-upon amount of money to the bank account provided by Y, but Y does not send the product (Figure 1). In the case of genuine fraud, the name and bank details provided by Y are most likely false, possibly belonging to a so-called "money mule". This is a person whose bank account is used for criminal activities, wittingly or otherwise.

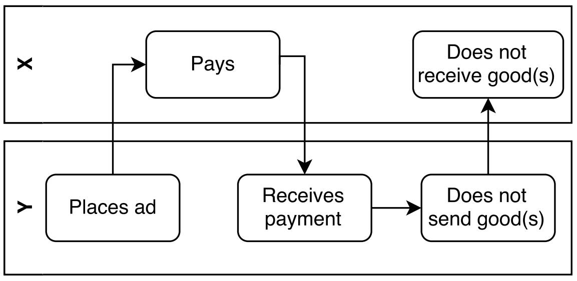


Figure 1. The classic swindle between victim X and swindler Y.

A more elaborate construction can be found in what may be translated as the "triangle swindle" which is depicted in Figure 2. From the point of view of the victim, there is no difference with the first classic scenario. However, the person to whom victim X's payment was transferred is not swindler Y, but rather person Z. Z received X's payment after having been contacted by Y about Z's ad on Marktplaats, after which he sent the goods to the address presented by Y. All this happened because Y copied Z's ad and allowed X to pay for the original ad. Not only has Y received this good for free, X believes that Z is the culprit and Z is not even aware of any complications until he is accused by X.

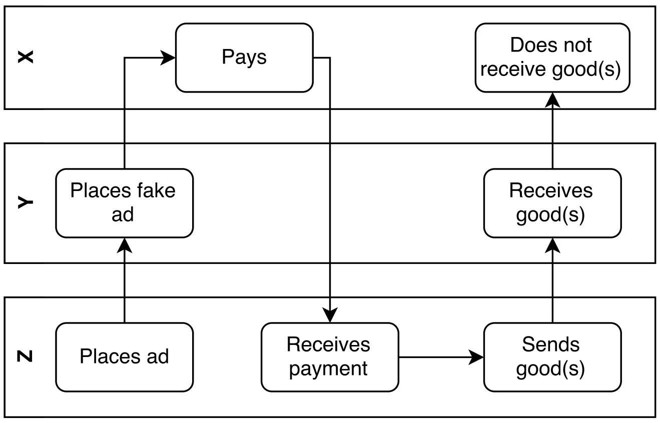


Figure 2. The triangle swindle between victim X, swindler Y and person Z.

Other possible scenarios do not involve trade sites such as Marktplaats, but involve more elaborate spoofed websites. Criminals create websites which seem almost identical to the original web shop they try to imitate. There are signs, of course, such as strange URLs and significantly lower prices. But some people will fall for this, genuinely believing that some big web shop has set up a discount version of their own shop. They are also persuaded by web shop certification marks, even though these are just images freely available on the internet. These spoofed websites allow for large groups of people to be scammed at once, by not sending them their orders. There is a variety of this spoofed website scenario where the web shop is not even trying to imitate another. These websites are completely registered at an address and with the Chamber of Commerce. They can be contacted by telephone and have a registered owner (a money mule). All this attention to detail makes them appear even more trustworthy, so they result in high amounts payments from victims.

2.2 Online intake of Trade Fraud Complaints

The current method of submitting a complaint consists of filling out an online form with some basic information, such as the complainant's details, details about the good or service that the citizen tried to purchase and any available details about email addresses, aliases and bank accounts used by the suspect. Furthermore, the form also has a free-text box in which the complainant is asked to fill in "what happened?" The form's contents are submitted to the police. The complainant is notified and might be contacted at a later stage regarding a follow-up investigation.

At the National Service Centre E-Crime, human analysts further manually analyse those entities (bank accounts, email addresses) from complaints that are suspicious. For example, if a particular bank account pops up in multiple complaints, there might be a fraudster at work. The analysts then take this entity and all related information from the different complaints, and visualize this as a "cluster", a mind-map showing the relations between entities such as bank accounts, aliases, URLs, and so forth and the basis of such clusters, and an accompanying Excel file, the case is built. First, further evidence is gathered from, for example, banks, e-commerce websites and internet service providers. The original complainants in a case are also contacted and asked for more evidence (email conversations with the suspect, screenshots). On the basis of this evidence, one or more scenarios are constructed about exactly what type of fraud has taken place.

One of the problems of the intake and investigation process on trade fraud is that there is a disconnect between the online form that a complainant fills in (i.e. the intake), and the construction and analysis of scenarios based on evidence (i.e. the investigation). The complainant does not always know exactly which information the police or judiciary need to follow up on a case. For a fraud, the victim should have been misled by false contact details, deceptive tricks or an accumulation of lies [2] – these are legal terms for which it is not immediately apparent exactly what they mean or how they should be proved. For example, if a victim was convinced to pay because the suspect offered the lowest price, then this alone is not enough for a fraud case. However, if the suspect also imitated a trusted party, the chances of successfully convicting the suspect for fraud increase significantly.

Because of the various subtleties mentioned above, it is often not clear at the time of filing the complaint exactly what sort of fraud (if any) has been committed. This is less of a problem for the regular intake process: when a complaint is filed at the police station, the complainant can state what happened to a police officer, who can match the incident to known fraud scenarios and ask further questions to try to confirm which particular type of scenario is applicable for the complainant's case. The online form is static and not connected to any possible fraud scenarios (which are only constructed

[2] Article 326 of the Dutch Criminal code

after multiple manual analysis steps), so it is impossible to ask any questions at the time of the intake.

Our initial aim is twofold: first, we need to connect the intake and investigation processes, so that the complainant can describe the incident and the system will directly structure it into scenarios and try to match it to known scenarios. Second, we want to make the intake process more dynamic through dialogue interactions. The subtle details in the scenarios are important but hard to capture and maintain in a large decision tree. So what is needed is a dialogue system that is able to ask the right questions based on the details given by the victim and on the scenarios that might be applicable. There are different fields in artificial intelligence that offer solutions for aspects of this application: multi-agent dialogue systems, computational linguistics and argumentation theory.

3 An A.I. System for Dialogues About Trade Fraud Scenarios

In this section we explain the general architecture of the proposed application for intake and investigation, with a focus on the different solutions from artificial intelligence that we use. Note that the work is ongoing: at the moment, the general agent architecture (Figures 3 and 5), the dialogue interface (Section 3.1) and natural language processing module (Section 3.3) have been developed and connected to each other. For dialogue management (Section 3.2) we will need to tailor the generic framework DGEP by Bex et al. [4]. Scenario reasoning (Section 3.4 is currently limited to basic ontology queries, but will be expanded with more advanced reasoners for argumentation-based semantics in the future.

Our application is a good example of a hybrid system containing both sub-symbolic artificial intelligence for machine learning and language processing, and symbolic artificial intelligence to reason about reports and cases. The system as a whole implements a dialogue system [17], and thus captures the process of intake and investigation. There are two main types of users: complainants who file new criminal complaints, and the police who want to analyse reports and combine them into a case file (Dutch: *proces-verbaal*).

A high-level overview of the system is shown in Figure 3 (for a more detailed view that focuses on the Natural Language Processing System see Figure 5). The complainant and police interact with the system through a dialogue interface. This interface allows users to submit input, i.e. make dialogue moves, but also shows the status of the dialogue such as the open questions. Questions can be generated by both the complainant and the police, but will also originate from the system itself through the scenario reasoning module. The dialogue is managed by a dialogue manager that maintains the legal moves of the participants. The legality of a move for a participant is based on the participants' commitments in the dialogue (e.g. statements that were made). The maintenance of the commitments in a commitment store is also part of the dialogue manager and its details are explained in [4]. The natural language processing system is called upon in case a participant provides free-text input. This system also maintains a knowledge graph that is constructed throughout the dialogue (Section 3.3. The graph serves as input for the scenario reasoning module of the application which then, based on the status of what is known about the reporter's incident, asks extra questions and clarifications through the dialogue manager. Finally, the scenario reasoning module also provides the analysis of reports and cases to the police.

We have opted to design the natural language processing and scenario reasoning components of the application with the agent

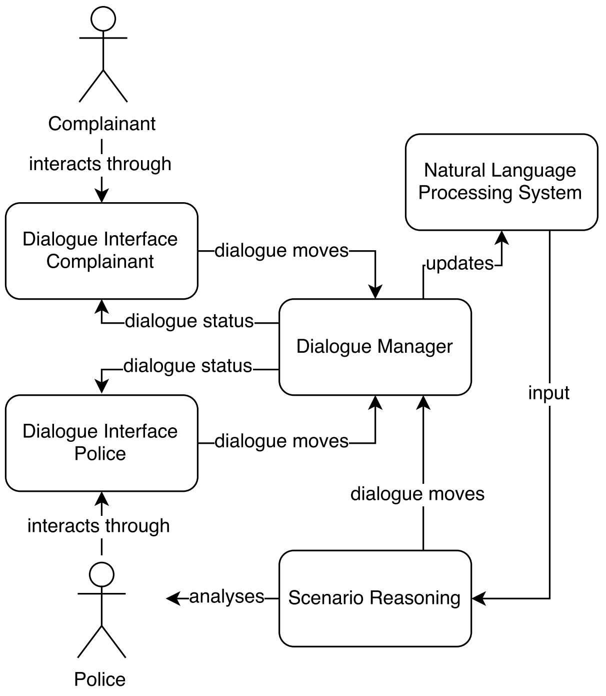


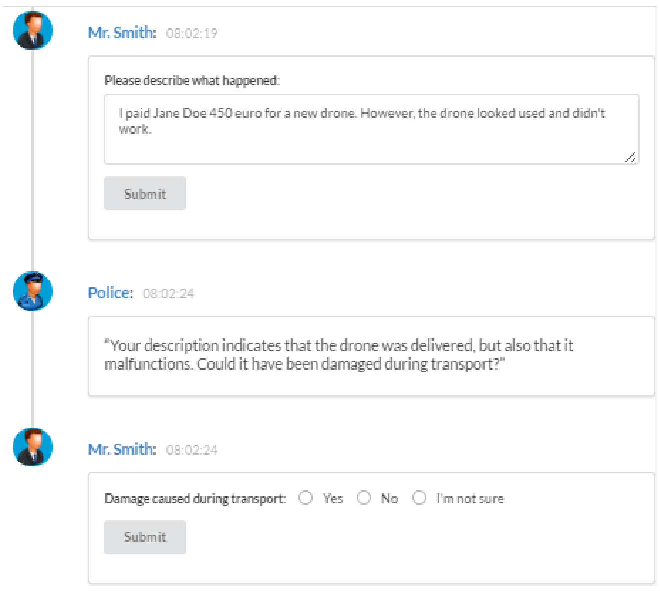
Figure 3. Architecture of the intake system. Boxes indicate software modules. Arrows indicate interaction between components such as service calls or input provision.

paradigm [21]. We use the object-oriented agent programming framework by Dastani and Testerink [10] to implement these components. This program is an object oriented translation of the logic-based programming language 2APL [9]. The agent paradigm matches modules of the software with high-level concepts such as beliefs, knowledge, goals and strategies. One of the main reasons for the agent paradigm is the dialogical nature of intake and investigation. It also benefits maintenance and eases the explanation of the software to outsiders. The agent paradigm also helps with our modularity goals as agents consists of modular components that implement their capabilities. The use of an object-oriented framework, in contrast to many logical approaches in agent oriented programming, not only further supports modularity but is also more accessible to programmers outside of academia. Finally, agent oriented software is distributed in nature, which accommodates the distribution of the application over a cluster of computers. We require such distribution for machine learning purposes, but also because the data is physically distributed and it is easier to attach an agent to data locally rather than collect all the data in a central location.

3.1 Dialogue Interfaces

Both the reporter of a crime as well as the police interact with the application through a dialogue interface. Figure 4 shows the current interface, in which a complainant, Mr. Smith, talks with a police agent, which may be a human or a software agent.

We recognize that natural language processing becomes increasingly more accurate, but also that sometimes it is easer for the application, or more comfortable for the user, to use a (partial) form. The dialogue that we envision will be a combination of free-text and forms, where it is often possible to switch between forms and free-text. For the interface layout we use basic web-based technologies. However, we also include speech-to-text (STT) and text-to-speech


Mr. Smith: 08:02:19

Please describe what happened:

I paid Jane Doe 450 euro for a new drone. However, the drone looked used and didn't work.

Submit

Police: 08:02:24

"Your description indicates that the drone was delivered, but also that it malfunctions. Could it have been damaged during transport?"

Mr. Smith: 08:02:24

Damage caused during transport: ○ Yes ○ No ○ I'm not sure

Submit

Figure 4. Screenshot of the dialogue interface

(TTS) solutions. These solutions are becoming more widely available and are getting increasingly more accurate. Various companies have now released free-to-use STT and TTS software for mobile devices. As part of future work, we want to investigate whether a keyboard-free, or perhaps even screen-free, dialogue system improves the user experience over a regular web interface. However, we focus on regular input for now as these technologies do not yet approach the accuracy which we feel will support a comfortable experience.

3.2 Dialogue Management

The dialogue that a complainant has with our application is not a completely free-text dialogue, but a mix between text and small forms. Also the topic of the conversation is strongly restricted to the task of reporting a crime which is well defined. This factor greatly reduces the amount of uncertainty that is encountered during the dialogue. Hence, we do not require a statistical approach to dialogue management (e.g. by modelling the dialogue as a Markov Decision process [16]).

The main aim of the system is to provide analyses of reports in order to support cases against swindlers. Because the analysis of scenarios and evidence can be naturally rendered in an argumentation formalism [1], our main approach to dialogue management comes therefore from argumentation dialogue systems theory. We take as a starting point the Dialogue Game Execution Platform (DGEP [4]). This platform allow us to specify the 'rules of the dialogue', such as turn taking, and then provides a mechanisms that keeps track of the commitments of a user. For instance, if a user states that he/she paid a particular seller, then we may commit that user to providing an argument why that particular seller was chosen over other alternatives. We note that not all information can be obtained from the user, hence some commitments are up to the police to satisfy. An example of this is the identifying information of a bank account which is one of the requirements to argue that a swindler used a false identity. The input that the users provide is stored in a knowledge graph that is explained in more detail in Section 3.3.

3.3 Natural Language Processing

In the agent-oriented architecture, there are basically two main types of agents: agents that enhance user input with extra knowledge sources and an agent that reasons about the known facts regarding a report. Both of these agents use a central knowledge base or scenario blackboard. Figure 5 shows a more detailed system architecture. In this section we discuss the agents of the first type (agents that enrich the user's input, i.e., the classifier, ontological, parsing and lexicon agents).

The textual nature of crime reports requires us to address natural language processing: for scenario reasoning, the relevant scenarios and entities from a criminal complaint are needed in a structured form, so that scenarios can be matched to typical fraud scenarios and further reasoned about using the evidence in the case (section 3.4). The task of the natural language processing module is therefore to identify entities in a text (e.g. companies, individuals and products) and then find the expressed relations between them (e.g. person A swindled person B, website C imitates company D).

We focus on (semi-)supervised techniques, where hand-crafted knowledge engineering is part of the design. Knowledge engineering comes in the form of ontology design (based on description logic) to specify the types of entities that we are interested in and the possible relations among them. A scenario model is an ontology plus extra instances of relations and concepts (the identified entities in a specific incident). There exist various frameworks for developing and reasoning with ontologies such as Protégé[3]. The widespread availability of triple storage and query technologies for ontology systems (such as Fuseki[4]) allows us to straightforwardly create a black board where agents can add and retrieve data. While it is challenging in general to define an ontology that covers all the necessary concepts, we have the advantage of having a specific domain: our ontology can be built using domain experts, existing crime reports and judicial documents.

The input from the dialogue interfaces is turned into a directed labeled graph called a *knowledge graph* (Figure 6). The graph combines the different knowledge sources such as the ontology for fraud cases that we use. Hence entities are adopted as nodes in the graph, but also other types of nodes exist, such as the words in the text input in the dialogue interface. The edges represent relations among the nodes, where labels of edges identify the type of relation. The objective of our natural language processing module is to predict new edges among entities in the knowledge graph. We illustrate our approach by assuming we have just the following two sentences.

1. "*I paid John*"
2. "*I paid no attention to the URL*"

Given these sentences we want to predict whether some type of payment has taken place. We do this by predicting whether a pay-edge exists between two identified entities. This is the case in the first sentence where the reporter pays a person named John. The final knowledge graph given the two sentences text is given in Figure 6, which we shall construct throughout the rest of this section.

The dialogue manager initiates the knowledge graph by inserting the words of the user as nodes in the graph. The parser agent then relates these words to each other by using a dependency parser – in our case, the Alpino parser for Dutch [8]. In our example graph, we use outgoing edges only for head dependents (i.e. 'paid', 'attention', 'to' and 'URL' are heads of (sub)dependency trees). A label 'X/Y' between words indicates that the dependency is of type 'X' and the

[3] http://protege.standford.edu
[4] https://jena.apache.org/index.html

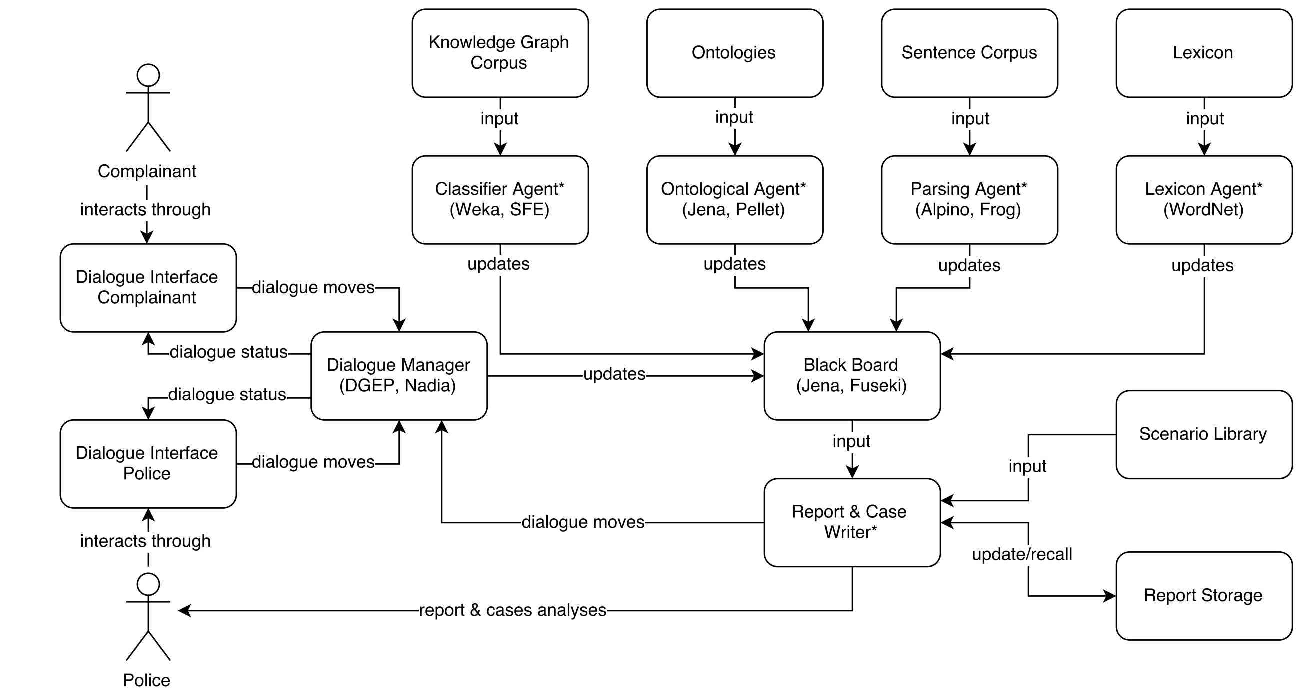


Figure 5. A more detailed system overview of the natural language processing module. Between parenthesis are possible alternatives to support the implementation of a system component. Components with an asterisk are implemented as an agent module and can be instantiated as independent agents or be combined in a single agent.

referred to word is on top of a syntactical tree of type Y. For example, in the second sentence there is an edge from 'paid' to 'to' labeled 'obj2/pp'. This indicates that 'obj2' is the name of the dependency relation and 'to' is on top of a prepositional phrase.

A lexicon agent can further to annotate the words with data from a dictionary in parallel to the parser agent. For instance, WordNet can be used for this [18], or in our case its Dutch variant by Postma et al. [19]. Using dictionaries to annotate words provides some foundational contextualisation and topicalisation of the text. For instance we may adopt the direct hypernym relations from WordNet, which group words in a semantic class. The direct hypernyms of 'pay', 'attention' and 'URL' are 'give', 'basic cognitive process' and 'address', respectively, according to WordNet. We represent this with labels 'X/Y' towards the WordNet node '_wordnet' where X is the relation that we extract ('dh' for direct hypernym) and Y is the word for that relation. Aside from WordNet, we may also use other sources

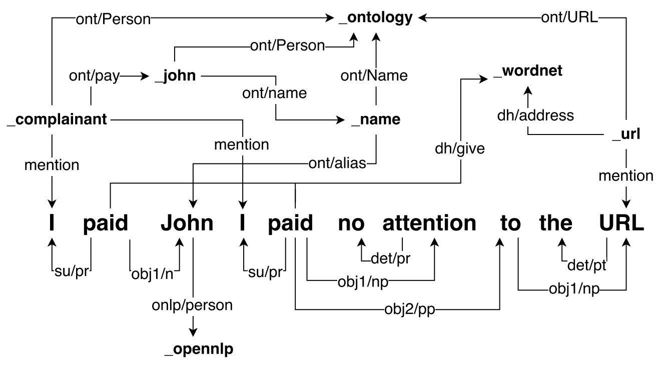


Figure 6. Example knowledge graph.

to classify words such as personal name and organisation name recognition tools. Such a tool would identify 'John' as a person's name, rather than a synonym for toilet as WordNet would classify 'John'. Since we are interested in name recognition, we do make use of this. OpenNLP [5] provides models for finding the names for persons, locations and organisations for both English and Dutch. We adopt the OpenNLP module as a node '_opennlp' and notate the classification of 'John' with the relation 'onlp/person' to indicate that OpenNLP classified 'John' as the name of a person.

The words and relations among those words form the foundation of a special classifier agent that identifies the entities in the sentences. Typically, these are the proper nouns and noun phrases. For the example text we identify the reporter who refers to himself/herself as 'I', we identify an entity named John, and we identify a URL. We adopt these entities as nodes in the knowledge graph, and connect their mentions in the text with a relation named 'mention'. The identification of entities and their mentions throughout a text is a combination of entity resolution and co-reference.

The identified entities in the text are the entities for which we want to determine how they are related to each other and how they are related to implicit other entities (which are not mentioned explicitly in the text). For this we apply classifier agents and ontological agents. Though some of these agents can operate in parallel, most of them have to be iteratively applied in order, because an update of one agent may trigger updates from another agent. Assume we have two disjoint ontological concepts for persons and for URLs. We want to represent in the knowledge graph that '_complainant' and '_john' are instances of persons, and that '_url' is an instance of URL. We do this by inserting the ontology that we use as a node '_ontology' in the knowledge graph, and connect the instances to this ontology by using the ontology's classification as a label edge, that is, '_john' is

[5] https://opennlp.apache.org

connected to '_ontology' with a label 'ont/Person'.

The process of determining the classification of ontological concepts and relations is part of the natural language processing module. We may use a different classifier for each relation that we want to predict. Some of these can be hand crafted. For instance, if an entity has a path with labels 'mention-onlp/person' towards the node '_opennlp', then we may predict that that entity has an 'ont/Person' labeled edge towards the '_ontology' node. Also, if the ontology specifies that a person has a name, then we can insert a name node ('_name') and relate that name to the person entity. Furthermore, a name has an alias. We can assume that an entity that has a name is mentioned by that name, using strings that OpenNLP classifies as person names. For instance, from a '_name' node we may follow all the possible paths ont/name$^-$-mention-onlp/person (superscript hyphen indicates a reverse relation), and then connect the second node of that path (the node that is connected by the mention relation) as an alias to the name. In our example, we identify this way that '_name' has an alias 'John'. Which is the name of the entity '_john'.

However, we may not always have the capability to hand craft classifiers. Furthermore, this may take up a lot of time and is hard to do for a large number of relations. Therefore, we have also looked into possibilities to learn classifiers using machine learning solutions. There are different techniques for learning edge prediction. We have opted for subgraph feature extraction by Gardner et al [13] to determine for an ontological relation what kind of path features (sequences of labels of paths) are predictive of an ontological relation. Consider we want to predict a pay edge between two entities from the text such as '_reporter' and '_john'. A one-side feature is a path feature that does not contain both nodes for which an edge is being predicted. For instance '_reporter' satisfies a path type mention-su/pr$^-$-dh/give. Subgraph feature extraction may find some good features such as '_reporter' satisfies mention-su/pr$^-$-obj/n-mention$^-$-ont/Person (meaning that the entity '_reporter' is mentioned as a subject in a sentence where the object is a person). Given these features and a corpus of example data, we can train classifiers such as support vector machines, or neural networks. We will make use of Weka [14] to actually train the classifiers.

At some point the classifier and ontological agents will signal that they do not have any more information to submit. Then, the agent that interacts with the user will evaluate the current graph and determine whether any further clarification from the user is needed in order to get a complete picture from the incident. The decision about what questions to ask is strongly based on an analysis of what scenarios can currently be constructed. This in turn is also part of the functionality of the scenario reasoning agent that will store the final graph and scenario. Hence these agents overlap quite strongly. If the user is asked for new input, then this input is added to the knowledge graph. Either free text is used, in which case the parsing and lexicon agents have to initiate again, or a form is used, in which case entities and relations can be directly added, and only the classifying agents have to be initiated again.

Finally we note that not all edges and entities can be added through automated means. For instance bank account information or user information from trading websites have to be obtained from third parties. Therefore, there is an interface for the police as well which can be used to add such information to the graph. We expect this information to be added only after the report is filed.

3.4 Scenario Reasoning

Once the information that comes in via the dialogue interfaces is included in the knowledge graph, it will become possible for the scenario reasoning agent to reason with this information. Multiple scenario reasoning agents can participate in a dialogue, using archetypical fraud scenarios from the scenario library and the repository of crime reports. The goal of such agents might be to, for example, match scenarios to typical fraud scenarios, compare scenarios given the available evidence, and elicitate further information from the user. These types of reasoning have been discussed in earlier work on the hybrid theory of scenarios and evidential arguments [6, 7, 1, 28], they have not been implemented. In this section we discuss briefly how implementations of the hybrid theory could be integrated in our system.

Figure 6 represents a (part of a) scenario as a knowledge graph. This knowledge graph can also be rendered as a short scenario, a sequence of events, with supporting evidential arguments, as usually presented in the hybrid theory (Figure 7; note that this figure is not itself a knowledge graph, but rather a more informal rendering of such a knowledge graph aimed at readability). Some elements of the scenario and its supporting arguments can be directly extracted from the knowledge graph: that 'C paid John' and that 'C paid no attention to the URL' can be directly inferred from the knowledge graph based on the complainant's report. The other elements – that 'John built a fake website' and that 'C believed the website was real' will have to be inferred from this (e.g. by applying scenario schemes or by asking the complainant in a dialogue, see below).

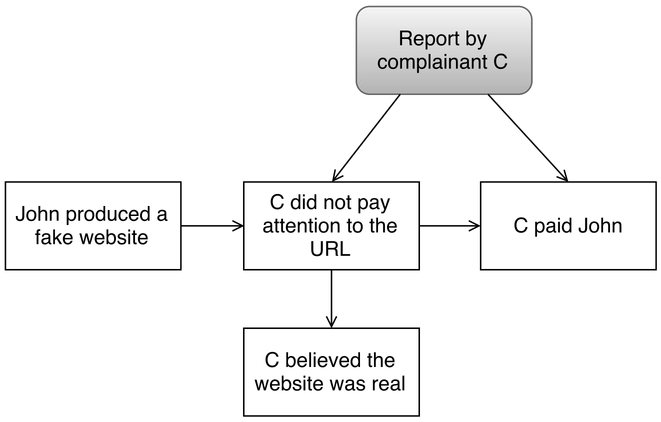

Report by complainant C
John produced a fake website
C did not pay attention to the URL
C paid John
C believed the website was real

Figure 7. A simple example of a scenario where white boxes represent scenario elements and the grey box an argument (evidence provided by the crime report).

One of the functionalities of the scenario reasoner will be to match the scenario posed by the complainant to typical fraud scenarios known to the police. In other words, it can be checked whether a scenario matches a *scenario scheme* [28]. These scenario schemes can be provided by police experts and are part of the scenario library. Matching knowledge graphs to scenario schemes can be kept relatively straightforward at first. With a few simple rules concerning the presence of entities and relations, specific fraud scenarios can be excluded. Without a mention of a website, for instance, a spoofed website scenario would be a nonsensical.

Excluding scenarios that a complaint does not describe is not necessarily the same thing as identifying the scenario that it does. Often, the complainant cannot distinguish between one scenario and an-

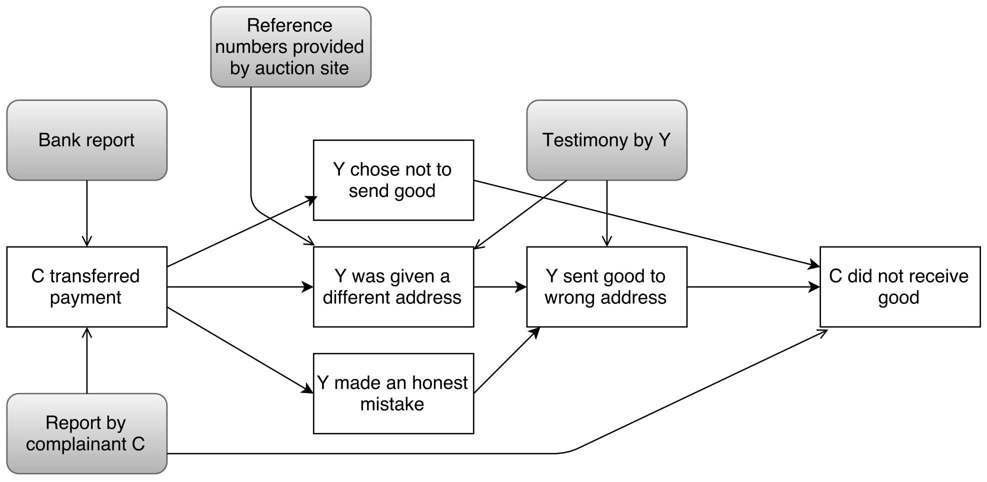


Figure 8. An illustration of one scenario being favoured over another due to an argument provided by the auction site.

other based on their perspective. The triangle swindle, for example, is intended to be indistinguishable from the classic swindle. Depending on the performance of such a rule-based exclusion and keeping in mind the possible future generalisation to other crime types, a more sophisticated solution may be worthwhile. This could be viewed as an exact graph matching problem in a unidirectional graph with labelled nodes and edges.

Determining the exact type of scenario often requires extra evidence. Investigation then becomes a process of inference to the best explanation: there are various possible scenarios that explain why the complainant did not receive the goods, but only one of them is the "true" scenario. Take, for example, the three possible scenarios in Figure 8. The complaint filed online only states that the complainant transferred the payment and did not receive the goods. There are then various explanations for this: the seller Y accidentally sent the goods to the wrong address, or the seller Y chose not to send the goods (a classic scenario, Figure1), or the seller was given a different address by swindler Z (a triangle scenario, Figure 2). Now, if Y testifies that he got the wrong address, and this is backed up by the auction website, chances are that we are dealing with a triangle scenario - of the three possibilities, the triangle option has the most supporting evidence.

The scenario reasoning agent also takes part in the dialogue. During the dialogue, the scenario model can thus lead to a question to be asked. Take, as an example, figure 7. Here, it is not explicitly mentioned by C that he never received the goods - if he would, there would not be a fraud case. From the typical scenario schemes it follows that in a fraud case, the victim should not have received any goods (or the wrong goods). So the scenario reasoning agent can ask the complainant whether they actually received the goods if the complainant did not mention this in first instance. Similarly, the scenario reasoning agent can ask the police analysts for extra evidence. In the situation of Figure 2, for example, the system can ask the police to contact the auction site and Y after the initial complaint, to see whether Y was given the right address. Thus, there are various ways to engage in dialogue about scenarios and arguments [6].

The scenario reasoning agent can ultimately reason with more than just the information from a single complaint. Very often, an investigation incorporates several complaints, it is not uncommon for criminals to be guilty of several types of crime, often even reflecting an overall strategy. The bank account numbers obtained through swindle may be used in another, for example. The hybrid theory allows for reasoning with more and larger cases simultaneously, even if they contradict one another. A combination of crimes is by no means necessarily restricted to trade fraud, as evidenced by the fact that criminals are often identified by linking them with cases from other police divisions. When these cases are themselves linked, such as when a money mule of a type of fraud reports someone for stealing is bank details, this will be reflected in the overall knowledge graph.

For the reasoning about scenarios and arguments to be incorporated into our system, we need to extend our ontology and scenario library with information about arguments - for example, what are the common ways in which a typical scenario or argument can be attacked or extended? Part of this ontology is already captured in the AIF ontology [20], which contains many argumentation schemes and associated critical questions and is available in various common formats (e.g. OWL, RDFs). Another element that the must be captured are the formal semantics of scenarios and arguments [7]. Again, the AIF ontology would be a good fit: as was shown in [5], the status of arguments expressed in the AIF ontology language can be determined using the common argumentation semantics that also underlie the hybrid theory [1].

4 Conclusion

In this paper, we propose a system in which several A.I. techniques are connected. Our system will the police to engage in a dialogue with crime reporters and use the resulting information to their advantage in the subsequent investigation. The system uses sub-symbolic techniques for machine learning and natural language processing to extract a knowledge graph from a complainant's scenario about what happened in a case, and then uses symbolic techniques such as ontologies and argumentative inference to reason with the scenarios and evidence contained in this knowledge graph. Furthermore, the reasoning and processing all takes place in a multi-agent architecture, which allows for modular system development, and is a natural fit with the dialogue interactions and interfaces. Our structured framework of scenarios and evidence establishes the foundation upon which formal reasoning can be applied, and can be used to connect multiple types of police data, which is in line with recent developments surrounding digital filing within the Dutch National Police.

The combination of natural dialogues and structured knowledge graphs will allow us to, for the first time, quickly and relatively simply build and reason about large cases. In the future this will allow for empirical assessment of the various formalisms designed to support evidential reasoning, as the textual dialogue interface allows users with little knowledge of these formalisms to understand and reason about the information in a case. Furthermore, the dialogue interface can also be used for knowledge elicitation. For example, there might be instances in which the classifier cannot accurately predict what type an entity is, or whether there is a relation between entities. The system can then ask a police analyst what the right type or relation should be in that case, and thus extend the ontology. Finally, in the future we also intend to incorporate text generation agents, which will be able to render parts of a knowledge graph as simple textual scenarios.

The techniques developed for our system are generalizable beyond the domain of online trade fraud. The idea of a linked data knowledge graph consisting of scenarios and evidence is applicable to many situations in which the police or judiciary reason with evidence. Two examples are risk assessment surrounding large events [3] and the assessment of asylum applications [2]. Extending the system to other domains will involve a substantial knowledge engineering effort, as scenario libraries will have to be built for different domains (e.g. scenarios surrounding football fan violence [3]. It is further possible to reason with more generic scenarios, such as the 'motivated action' scheme [28] - there are many ontologies available that allow for reasoning with events, time, arguments, and so forth. Finally, we are currently performing data analysis on police data surrounding online crime, which might lead to novel scenarios, frequent patterns that do not correspond to any known scenarios. For example, given the data we have in our project we can try to determine and validate which types of complaints are usually withdrawn (usually because goods have been delivered after all), or designated as being civil rather than criminal (e.g. the delivery of a damaged item or one that is a cheap copy).

ACKNOWLEDGEMENTS

This research is funded by the Dutch National Police Innovation Programme.

References

[1] Floris Bex, 'An integrated theory of causal stories and evidential arguments', in *Proceedings of the 15th International Conference on Artificial Intelligence and Law*, pp. 13–22. ACM, (2015).

[2] Floris Bex and Viola Bex-Reimert, 'Evidence assessment in refugee law with stories and arguments', *Informal Logic*, (2016). to appear.

[3] Floris Bex and Bas Hovestad, 'An argumentative-narrative risk assessment model', in *Proceedings of the European Intelligence and Security Informatics Conference (EISIC) 2016*, (2016). to appear.

[4] Floris Bex, John Lawrence, and Chris Reed, 'Generalising argument dialogue with the dialogue game execution platform', 141–152, (2014).

[5] Floris Bex, Sanjay Modgil, Henry Prakken, and Chris Reed, 'On logical reifications of the argument interchange format', *Journal of Logic and Computation*, **23**(5), (2013).

[6] Floris Bex and Henry Prakken, 'Investigating stories in a formal dialogue game', in *Proceedings of the 2008 conference on Computational Models of Argument (COMMA 2008)*, pp. 73–84. IOS Press, (2008).

[7] Floris J. Bex, Peter J. van Koppen, Henry Prakken, and Bart Verheij, 'A hybrid formal theory of arguments, stories and criminal evidence', *Artificial Intelligence and Law*, **18**(2), 123–152, (July 2010).

[8] Gosse Bouma, Gertjan Van Noord, and Robert Malouf, 'Alpino: Wide-coverage computational analysis of dutch', *Language and Computers*, **37**(1), 45–59, (2001).

[9] Mehdi Dastani, '2APL: a practical agent programming language', *Autonomous Agents and Multi-Agent Systems*, **16**, 214–248, (2008).

[10] Mehdi Dastani and Bas Testerink, 'From multi-agent programming to object oriented design patterns', in *Engineering Multi-Agent Systems*, 204–226, Springer International Publishing, (2014).

[11] CJ de Poot, RJ Bokhorst, Peter J van Koppen, and ER Muller, *Rechercheportret - Over dilemma's in de opsporing*, 2004.

[12] Norman Fenton and Martin Neil, *Risk assessment and decision analysis with Bayesian networks*, CRC Press, 2012.

[13] Matt Gardner and Tom Mitchell, 'Efficient and expressive knowledge base completion using subgraph feature extraction', *Proceedings of EMNLP. Association for Computational Linguistics*, **3**, (2015).

[14] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten, 'The WEKA Data Mining Software: An Update', *SIGKDD Explor. Newsl.*, **11**(1), 10–18, (November 2009).

[15] John R Josephson, 'On the proof dynamics of inference to the best explanation', *Cardozo L. Rev.*, **22**, 1621, (2000).

[16] Esther Levin, Roberto Pieraccini, and Wieland Eckert, 'Using markov decision process for learning dialogue strategies', in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 1, pp. 201–204. IEEE, (1998).

[17] Peter Mcburney and Simon Parsons, 'Dialogue Games for Agent Argumentation', in *Argumentation in Artificial Intelligence*, eds., Iyad Rahwan and Guillermo Simari, chapter 22, 261–280, Springer, (2009).

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

[19] Marten Postma and Piek Vossen, 'Open source dutch wordnet', (2014).

[20] Iyad Rahwan, 'Mass argumentation and the semantic web', *Web Semantics*, **6**(1), 29–37, (feb 2008).

[21] Yoav Shoham, 'Agent-oriented programming', *Artificial intelligence*, **60**(1), 51–92, (1993).

[22] Franco Taroni, Colin Aitken, Paolo Garbolino, and Alex Biedermann, *Bayesian Networks and Probabilistic Inference in Forensic Science*, Wiley, Chichester, 2006.

[23] Sjoerd T Timmer, John-Jules Ch Meyer, Henry Prakken, Silja Renooij, and Bart Verheij, 'Explaining bayesian networks using argumentation', in *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pp. 83–92. Springer, (2015).

[24] Alice Toniolo, Timothy J Norman, Anthony Etuk, Federico Cerutti, Robin Wentao Ouyang, Mani Srivastava, Nir Oren, Timothy Dropps, John A Allen, and Paul Sullivan, 'Supporting reasoning with different types of evidence in intelligence analysis', in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pp. 781–789. International Foundation for Autonomous Agents and Multiagent Systems, (2015).

[25] Alice Toniolo, Timothy J Norman, and Katia P Sycara, 'An empirical study of argumentation schemes in deliberative dialogue', in *Proceedings of ECAI 2012*. IOS Press, (2012).

[26] Susan W van den Braak, *Sensemaking software for crime analysis*, Ph.D. dissertation, Utrecht University, 2010.

[27] PJ van Koppen, *Overtuigend bewijs: Indammen van rechterlijke dwalingen*, Nieuw Amsterdam, Amsterdam, 2011.

[28] Bart Verheij, Floris Bex, Sjoerd Timmer, Charlotte Vlek, John-Jules Meyer, Silja Renooij, and Henry Prakken, 'Arguments, scenarios and probabilities: connections between three normative frameworks for evidential reasoning', *Law, Probability & Risk*, **15**, 35–70, (2016).

[29] Charlotte S Vlek, Henry Prakken, Silja Renooij, and Bart Verheij, 'Building bayesian networks for legal evidence with narratives: a case study evaluation', *Artificial Intelligence and Law*, **22**(4), 375–421, (2014).