Implementing Soft Fusion

Holger Köhler, Dale A. Lambert, Jan Richter, Glenn Burgess, Tim Cawley
Intelligence Processing and Analysis Branch
Defence Science and Technology Organisation
Adelaide, Australia

Abstract—This paper outlines an implemented process for fusing soft textual information with hard signal information called the Mixed Initiative Soft Fusion Implementation Testbed (MISFIT). While translation of the content of hard information sources to a machine processable semantic representation is usually feasible, the content of most unconstrained natural language textual sources can only be partially translated to a machine processable semantic representation. In response the Intelligence Processing and Analysis Branch has differentiated four alternative strategies. This paper describes the mixed initiative strategy in which “humans in the loop” address components of the process that are difficult to automate. This strategy increases potential for complex semantics to be expressed, structured and exploited with machine reasoning.

Keywords—hard and soft fusion; semantic representation; automated reasoning; big data; mixed initiative.

I. INTRODUCTION

This paper outlines an implemented process for fusing soft textual information with hard signal and image information. The adopted multi-source fusion framework is illustrated in Fig. 1, taken from [1]. It applies the JDL fusion levels (e.g. [2]) 0 to 3 to signal, image and textual sources. Under this framework, lower-level fusion (levels 0 and 1) is conducted for each source type, with multi-source fusion illustrated in Fig. 1 by speech to text (S2T) and optical character recognition (OCR), while higher-level fusion (levels 2 and 3) operates with all three source types being translated into a canonical machine processable semantic representation. Each of the elements in Fig. 1 is an applied instance of the State Transition Data Fusion Model (STDF) ([1]).

Automated STDF processes for hard signal and soft textual fusion at each JDL level are described in [1]. While translation of the content of hard information sources to a machine processable semantic representation is usually feasible, the content of most unconstrained natural language textual sources can only be partially translated to a machine processable semantic representation. In response the Intelligence Processing and Analysis Branch has explored four alternative strategies to deal with text content, outlined in section II. The remainder of the paper then develops one of these strategies and applies it to the fusion of information relating to the entities, attributes and relationships of a social network. Based on the Universal Process Guide for Social Network Analysis (UPG) framework of [3], the paper presents a four stage soft fusion process comprising: Collation (section III); Coding (section IV); Cleaning (section V); and Analysis (section VI) stages. Section 0 then delivers the conclusion for the paper.

II. SOFT FUSION STRATEGIES

Douglas [11] identifies three problems for data management: large volumes, high velocity and wide variety. Veracity is also often included to provide the four “V”s. These problems are especially applicable when fusing the content of unconstrained natural language textual sources of information and can be addressed using four distinct soft fusion strategies summarised in Fig. 2.

![Fig. 2. Comparison of soft fusion strategies](image)

Traditionally information retrieval systems are applied to the problem where the challenge is to build a search index containing enough information to retrieve all documents that match a given search query. This traditional information retrieval strategy is document focused. Machine processing is applied to assign, store and generate metadata about a text document to assist the machine to retrieve relevant documents from a wider corpus. Efforts to use the machine to assist in
understanding a document’s semantic content are limited to generation of metadata. Such a generated machine processable representation of a document is not designed to enable reasoning but rather to enable retrieval. The fusion, deep understanding and analysis of the unstructured text content is left up to the human reader. This leads to high quality results, but does not scale as the volume of information grows. Increasing the number of humans interpreting documents does not speed up situation assessment as each human needs to build the relevant semantic structures in their mind without being able to easily share them.

The second strategy is a mixed initiative strategy in which “humans in the loop” (HIL) address components of text processing that are difficult to automate, such as creating meaningful structured representations of text content. Machine processing is applied in the analysis of a structured representation of relevant text content to reduce the time taken and cognitive load of analysing large, complex datasets. The machine analysis requires structured data, but generating and validating such structure from text requires investment of human time. The human is relied on heavily to ensure that the structured data is of high quality and suitable for the analytical questions being investigated. Volumes of relevant text need to be low, otherwise investment of human time to create a suitable structure would be prohibitive. In this strategy technologies such as information extraction are applied, but are considered too inaccurate to use without human validation of their results. This strategy increases the potential for complex semantics to be expressed, structured and exploited with machine reasoning due to the HIL.

The third fully automatic strategy focuses on machine processing without the HIL. Processing includes information extraction, entity resolution, information management and knowledge representation and reasoning. This strategy relies on information extraction to deal with the initial problem of converting from text content to machine processable information. Information extraction results in a partial and shallow machine representation of text content. This strategy involves consuming the output of one machine process as the input into the next and can therefore lead to a compounding of errors. Due to such errors it is suggested that such processing is best suited to gisting and alerting functions where results need to meet an interestingness measure before they are raised for human inspection and verification. Such a strategy runs the risk of producing an excessively low signal to noise ratio that could frustrate the human consumers of the machine generated insights. Care should be taken about how such a strategy is applied.

The fourth controlled natural language (CNL) strategy seeks to constrain the language used when creating text documents, to language which can be automatically translated into a machine processable semantic representation. This strategy is only feasible for information sources within organisational control and therefore can’t be the only strategy applied.

The remainder of this paper describes an implemented Mixed Initiative Soft Fusion Implementation Testbed (MISFIT) process for following the mixed initiative strategy as shown in Fig. 3.

![Fig. 3. Mixed Initiative Soft Fusion Implementation Testbed (MISFIT)](image)

### III. Collation

The goal of Collation is to identify a set of relevant text documents that are worth investing human effort in structuring during the subsequent Coding stage [3]. The mixed initiative strategy builds on the traditional strategy by applying additional machine processing in the Collation stage to improve the human’s ability to choose the best set of documents on which to spend their valuable time.

The Collation stage includes ingestion, translation, content indexing, information retrieval, notification and document triage.

#### A. Ingestion and Translation

Ingestion involves format identification, format transformation, language identification, language translation, text normalisation and information extraction. To make the best use of the semi-structured nature of many text sources, these processes need customisation. Customising the information extraction processing applied to individual data sources enables the extraction of entities, attributes and links that are more relevant to the desired analysis. Although information extraction systems can be tuned, the deployment of such advanced text processing systems requires adequate ongoing expert human support and a thorough knowledge of the information requirements. In the authors’ experience, most organisations have a poor understanding of their information requirements, the capabilities offered by such advanced tools, and the need to change their current business process to include such support, thus resulting in the deployment of poorly customised extraction capabilities.

Machine translation of foreign language texts is an effective aid, especially for gisting. HIL comparison, verification and correction improve the effectiveness of machine translation markedly [12].
B. Content Indexing

Information extraction technology can be applied to enhance the document indexing, search and retrieval processes. Even a low accuracy information extraction system still provides useful metadata for information retrieval purposes because the cost of a false positive is lower when results are being read by human rather than machine, as occurs when applying the fully automatic strategy. Additional processing, such as geocoding place names into spatial coordinates, can provide additional information to add to the index.

The MISFIT process uses Apache Solr services over Apache Lucene index technology. Lucene indexes are able to represent very simple data structures and are mostly used to describe documents (i.e. metadata) using simple name value pair fields. Big Data technologies such as Apache Hadoop are being investigated for use in conjunction with Lucene and Solr to address scalability.

In the traditional strategy, text is indexed on words used in content and correct search results contain search terms that are explicitly found in the text (apart from simple features such as synonym lookups and fuzzy string matching). Semantic searches aim to retrieve documents based on search criteria that include words and concepts that are not necessarily explicitly mentioned in the text. This enables the retrieval of relevant documents that would otherwise be missed. Enabling semantics to be incorporated in the information retrieval system involves a combination of content processing, to annotate concepts of interest, formal modelling of semantic concepts together with the application of such semantic models in the search systems. The same ontology that is required to facilitate the cleaning and automated reasoning described later can be applied during document indexing. Even a weakly expressive ontology can be useful as long as there is some level of match with the concepts required for automated reasoning.

The mixed initiative strategy uses the same search techniques as the traditional strategy, but adds additional document triage processes.

C. Document Triage

Document Triage is the process whereby a user is able to reduce the set of text documents of interest to a manageable level using a combination of iterative searches, filtering, notification services, corpus visualisations and prioritisation algorithms. Putting greater effort into choosing the right set of text documents, before annotating them to create a structured semantic form, allows HIL time to be used more efficiently. Machine-facilitated triage benefits from applying the information extraction processing during the document processing prior to the creation of search indexes, as well as post indexing where a potentially deeper information extraction system can be applied to the selected texts. Using information extraction, entity resolution and knowledge base population techniques allows for automatic creation of structured data. Such data is most appropriately applied to improving the document triage process by enabling a HIL to quickly determine whether further searches for documents or entities would be required to improve the relevance and coverage of the text documents being structured during the Coding stage.

IV. Coding

Coding refers to the process of turning text content into a structured form suitable for machine assisted, algorithmic analyses [3]. The Coding stage includes document markup and semantic representation.

A. Document Markup

The process of turning text content into a structured knowledgebase (KB) and combining it with known structured data is shown in Fig. 4. Manual document markup user interfaces allow a user to select spans of text and assign them as values of (and evidence of) a slot in a structured knowledgebase. This manual markup can be machine-assisted with information extraction tools to speed up the annotation process. Information extraction outputs can be shown as suggestions of potentially true, and potentially relevant concepts. The HIL can then manually accept or reject the machine's suggestions.

Fig. 4. Structuring text and fusing with known structure

An alternative method of document markup of uncontrolled text is to use CNL to annotate the natural language text. CNL annotation is a way to bridge the gap between user and machine by asking the user to summarise sections of relevant information content in a controlled language that can be read by humans like natural language. This is beneficial because CNL can have a pre-defined machine semantics whereas annotated natural language only has a partial machine semantics.

B. Semantic Representation

Conventional database and markup language systems are mediums through which we can store meaningful information so that ourselves or others can subsequently retrieve that information and understand its meaning. But like post office boxes, these machines have no understanding of the information they hold, and therefore offer a limited capacity to meaningfully assist. The significance of the term “semantic” in “semantic representation” is that it signals the ambition to move fusion machines beyond a post office status. This involves enabling fusion machines to not just store and reproduce information, but to an extent understand the meanings of the stored information, and therefore respond far more effectively.
Machine based semantic representations are possible through the 8 steps detailed in [4]. In essence this involves the representations being expressed in a formal language $L$ for which there is an associated formal logic $<L, \vdash>$ with inference relation $\vdash$. Formal theories (sets of formal language sentences) are used to express truth conditions, which under a Davidsonian interpretation ([5]) of Tarskian truth ([6]), delivers the meaning of the formal language terms in the presence of logic $<L, \vdash>$. The machine is then able to automatically process these meanings if there is a sound and complete computational logic $<L, \vdash>$ that computes with formal language $L$ in accordance with logic $<L, \vdash>$.

There is a trade-off between the expressivity of any formal logic $<L, \vdash>$ and its computability. Description Logics (e.g. [7]) have sought weak expressivity to secure computability, while First-Order Logics, and those of greater expressivity, incur increasing degrees of undecidability ([8]). Within the MISFIT framework of Fig. 3, Description Logics are suited to the simpler semantic triage function during Collation, but a more expressive logic is required during Coding, Cleaning and Analysis. The WHY logic has been developed for the latter purpose (see section V.C), with expressivity slightly beyond First-Order Logic, by allowing the nesting of atomic propositions. If required, uncertainty can be handled through probability distributions over possible worlds, with possible worlds expressed as sets of WHY sentences (see [1]).

Having decided upon the WHY class of logics, the following step is to determine both the WHY formal language $L$ that will be used to describe the world of interest, and the formal theory $M$ that specifies the truth condition meanings for the terms of $L$. The Mephisto Semantic Representation ([9], [10]) has been employed for this purpose. Mephisto represents the world through the 5 layers: Metaphysical, which introduces concepts associated with identity, existence, time and space; Environmental, which characterises parts of the world environmentally in terms of land, water, air, et cetera; Functional, which classifies functions like sensing, striking, moving, carrying, informing, interpreting and transforming; Psychological, which specifies affective, behavioural and cognitive mental processes; and Social, which formulates notions like authority, conflict and parent.

The remaining issue is how to interface conventional database and markup language systems with the aforementioned semantic capability.

The conventional database and markup language system tools are generally not very expressive in their knowledge representations. While this promotes flexibility for the user, it also means there are usually no explicit formal semantics. To enable some machine reasoning in such systems, their data and ontology must be mapped and transformed, which could be a lossy process. This is much more likely in the reverse where information from a richer knowledge representation is passed back, possibly in answer to some specific reasoning needs. Fitting a less expressive knowledge representation into a more expressive one provides a less precise data input to a reasoning system, but fitting a more expressive knowledge representation into a less expressive one means that some things can’t be returned. Potentially this could be managed by linking between facts in the less expressive system and those in the more expressive system so that greater detail can be looked up as required.

In any case, there will need to be something more expressive to enable reasoning examples such as inferring potential meetings based on spatio-temporal collocation or inferring influence links based on a combination of kinship, authority, resource sharing and affect.

Palantir is a commercial knowledgebase and visual analytic system [13]. It is used as an example of conventional knowledgebase capability that has been extended to demonstrate concepts of soft fusion in the implementation of MISFIT.

A Palantir/Mephisto bridge was implemented to understand the issues involved in interfacing conventional knowledgebase representations with the Mephisto semantic capability. It continues to be an investigative platform for researching the resolution of ambiguity and incompleteness in the mapping process. A longer term research question to explore, is how to impose strict semantics on systems like Palantir to facilitate and strengthen the Mephisto interfacing.

Palantir’s core data structure is customisable via its flexible ontology. Multiple domain structures can be modelled and the ontologies for each domain can vary vastly. Palantir's flexibility allows concepts to be represented in multiple ways; for example a link can be represented by a simple edge between two objects, or via a dedicated link object which can be used to store a wealth of information about the link. Palantir objects can have date and location metadata associated with them, but the user must decide what each date and location means.

Data within the Palantir knowledge base is mappable to a Mephisto representation, allowing it to be ingested by Mephisto and combined with semantic information from other sources. This is done by targeted mapping logic, which consumes specific knowledgebase values in order to represent them correctly in Mephisto. The mapping from Palantir to Mephisto is essentially a non-enforceable semantic interpretation of Palantir data. Until semantics can be enforced on Palantir data, the mapping must be tailored to a specific ontology, as well as the chosen data representations. This ensures Mephisto is fed information that is as unambiguous and complete as possible, ensuring that the inference engine does not deliver wrong, potentially dangerous answers.

An example of transforming a Palantir person to a Mephisto representation is shown in TABLE I below. The translation is performed semi-automatically, with the user choosing Palantir objects to be translated, and then applying translation logic encapsulating chosen semantic mapping decisions. Palantir object identifiers are used to correlate queries with answers between the two systems.
The process of correcting errors in structured data is an important aspect of data cleaning. In many cases, errors in human-coded data can be significant, ranging from typographic errors or misclassified instances and false positives and negatives. It is crucial to perform some kind of belief resolution of other concepts. This can be implemented computationally intractable.

Typical errors will include: typographic errors, misclassified instances and false positives and negatives. It will also typically include errors of conceptual uncertainty – where users were inconsistent or unclear about how a concept described in the text would best be represented in the ontology, often because the writer was themselves unclear. In addition there will be errors of fact, which can potentially be corrected by finding contradictory facts and performing some kind of belief resolution. Data cleaning can also be used to improve the quality of the data by checking for internal consistency and by performing targeted searches of the wider corpus (or the internet) to look for specific pieces of information, such as a person’s date or place of birth.

For each concept of interest, there may be multiple, possibly conflicting, observations. A belief revision process is used to determine which of these observations should be used in the analysis. This may be as simple as choosing the ‘most likely’, but the best approach to use depends on the nature of the concept.

Data cleaning can also be used to improve the quality of the data by checking for internal consistency. Typically, concepts are interrelated, so observations of one concept can affect the belief resolution of other concepts. This can be implemented using backwards chaining, so the assessment of each concept instance applies a deductive reasoning process to determine all the relevant observations. However, this can be computationally intractable.

An alternative is to use forward chaining from observations to assessments. This process can be done as a batch, during a separate forward chaining phase or, incrementally, after each observation is added to the system. For example, if we have recorded that Alice and Bob are siblings, and then we learn that Carol is Alice’s sister-in-law, then this could mean that Carol is also Bob’s sister-in-law or that she is his spouse. Often the source material will not be more specific, but a human looking at the available information may be able to determine which of these cases is more likely (by looking at their surnames for instance). When manual resolution like this is required it may be best to do it at the time the piece of information that leads to the deduction is added, or shortly after, as the user will already be familiar with the situation.

B. Data Transformation

Another activity that occurs during data cleaning is a review of the available data holdings and an assessment of its value for analysis. This in turn may guide some steps of data cleaning to make the data more useful. For instance, suppose it is found that 80% of people have place of birth entered, but only 10% have the name of the town, 20% have the state and the rest have the country. It is going to be more useful for most analysis purposes to generate a ‘country of birth’ attribute or build this inference into the formal logic, since this is at a consistent scale and mostly populated.

The above is an example of generating attributes from other attributes. Other kinds of transformations include generating attributes from links (such as inferring a ‘married’ personal status variable from the presence of a linked ‘spouse’), links from attributes (say, converting a place of birth attribute into a link to a location) and links from other links. As an example of the latter, suppose information is coded for instances of a number of different types of interaction and relationship types.
between actors in a network. Individually, each of these link types is likely to be a sparse network – suppose there is a sample of emails, financial transactions and kinship links. Taken together it may be possible to say something more general about the relationships between dyads, such as their overall feelings towards each other – positive or negative. This will typically produce a denser network of less-specific links, which is useful for some kinds of analysis (such as looking for factions).

C. Automated Inference

Data cleaning is performed by the HIL with automated assistance. In its simplest form the automated assistance performs menial tasks like automatic spelling correction relative to a lexicon. More sophisticated automated assistance involves user access to automated reasoning across the assembled information. The WHY automated reasoner can be applied for this purpose. The following paragraph offers a simple illustration.

The WHY reasoner allows Mephisto definitions and axioms (formal theories) to be asserted through tell_define, tells_front and tells_rear operations. For example, asserting that a mother is a female parent, that a person is male or female, and that being a male precludes being a female, can be asserted in the following way:

```prolog
tell_define([source(mephisto)], mother(X, Y), female(X) & parent(X, Y)).
tells_front([source(agent1), confirmed], person(spinnaker) & ss_mole(spinnaker)).
tells_front([source(agent2)], mother(spinnaker, someone22) & lives(someone22, london)).
tells_front([source(agent3)], male(spinnaker)).
```

Suppose we additionally have information from three agents: confirmed information from agent 1 indicating that there is a security service mole named Spinnaker; agent 2 stating that Spinnaker is the mother of someone living in London; and agent 3 claiming that Spinnaker is a male:

```prolog
tells_front([source(agent1), confirmed], person(spinnaker) & ss_mole(spinnaker)).
tells_front([source(agent2)], mother(spinnaker, someone22) & lives(someone22, london)).
tells_front([source(agent3)], male(spinnaker)).
```

The WHY query language can query using satisfy, true, false, incomplete and inconsistent predicates, and bind variables in the process. For example, the following user query searches for any inconsistencies about males. It locates an inconsistency with Spinnaker, and finds no other male inconsistencies.

```prolog
| ?- inconsistent(male(X)).
| X = spinnaker ? ; no
```

Thus male(spinnaker) and its negation ~male(spinnaker) must both be deducible. The WHY reasoner’s automated explanation facility then allows the user to understand why each is the case and thereby decide upon an appropriate cleaning strategy. Of course the user is not exposed to this underlying logical representation, but is instead presented with a CNL translation of it or some equivalent explanation interface (see section VI.B).

```prolog
| ?- show_why(male(spinnaker)).
1. [male(spinnaker)] is derived from told([source(agent3)], male(spinnaker)) yes
| ?- show_why(~male(spinnaker)).
1. [~male(spinnaker) if (female(spinnaker))] is derived from told([source(mephisto)], all([_4174], (male(_4174) => ~female(_4174))))
2. [~male(spinnaker)] is derived from modus ponens by 1 and [female(spinnaker)] by 4
3. [female(spinnaker) if (mother(spinnaker,someone22))] is derived from told([source(mephisto)], define(mother(_4614, _4615), female(_4614 & parent(_4614, _4615))))
4. [female(spinnaker)] is derived from modus ponens by 3 and [mother(spinnaker,someone22)] by 5
5. [mother(spinnaker,someone22)] is derived from told([source(agent2)], (mother(spinnaker,someone22) & lives(someone22, london))) yes
```

VI. ANALYSIS

Analysis is a process of drawing conclusions and sometimes constructing and testing hypotheses about a system of interest using observations of the system.

Analysis typically involves some amount of ‘handle turning’ and some amount of inspiration. It also typically involves some amount of backtracking in the process – to infer new variables, to fix data errors that become apparent, and sometimes to recode data that is evidently wrong or missing. This tendency for backtracking highlights the importance of being able to automate and control as many of the ‘handle-turning’ steps as possible, including the Cleaning and Analysis stages. It also highlights the importance of being able to manage incremental updates to the data and a history of all decisions made previously, so that the HIL only needs to reconsider time-consuming decisions when the underlying data might affect the previous assessment.

A. CONTENT ANALYSIS

The automated inferencing mechanism outlined in section V.C can equally be applied to assist user analysis. Suppose cleaning resulted in the rejection of agent 3’s advice in the previous example. The analyst knows that Spinnaker is the only security service mole

```prolog
tells_front([source(analyst)], all(X), (~identical(X, spinnaker) => ~ss_mole(X))).
```

and that a female security service mole can only be one of three people: Brenda Rogers; Carlie Rhodes; or Sylvia Westcott.

```prolog
tells_front([source(analyst)], ss_mole(brenda_rogers) v ss_mole(carlie_rhodes) v ss_mole(sylvia_westcott)).
```
But agent 1 has confirmed that Spinnaker was in Rome on 7-Jul-13, while agent 2 established that Brenda Rogers was visiting her daughter in London at that time, and that Carlie Rhodes was in New York.

tells_front([source(agent1), confirmed], at(spinnaker, timestamp(2013, 6, 7, 10, 42, 12.1), rome)).
tells_front([source(agent2), confirmed], at(brenda_rogers, timestamp(2013, 6, 7, 10, 42, 12.1), london)).
tells_front([source(agent2), confirmed], at(carlie_rhodes, timestamp(2013, 6, 7, 10, 42, 12.1), new_york)).

The analyst adds the fact that two individuals at different locations at the same time cannot be identical,

tells_front([source(analyist)], all([X, Y, T, SX, SY], (at(X, T, SX) & at(Y, T, SY) & ~same_location(SX, SY)) => ~identical(X, Y))).

This then allows the WHY reasoner to deduce that Sylvia Westcott is Spinnaker.

| ?- satisfy(identical(X, spinnaker)).
X = sylvia_westcott ? ;

no

The reasoner’s generated explanation is pasted below:

| ?- show_why(identical(X, spinnaker)).
1. [identical(sylvia_westcott,spinnaker) if (~ss_mole(brenda_rogers) & ~ss_mole(carlie_rhodes)))] is derived from resolution with [~ss_mole(_13025) v identical(_13025,spinnaker)] by 11
2. [identical(sylvia_westcott,spinnaker)] is derived from modus ponens by 1 and [~ss_mole(brenda_rogers)] by 5 and [~ss_mole(carlie_rhodes)] by 9
3. [ss_mole(brenda_rogers) v ss_mole(carlie_rhodes)] is derived from told([source(analyist)], (ss_mole(brenda_rogers) v ss_mole(carlie_rhodes) v ss_mole(sylvia_westcott))

4. [~ss_mole(brenda_rogers) if (at(brenda_rogers,timestamp(2013,6,7,10,42,12.1), london) & at(spinnaker,timestamp(2013,6,7,10,42,12.1), rome) & ~same_location(london,rome)] is derived from resolution with [~ss_mole(_12798) v identical(_12798,spinnaker)] by 11
5. [~ss_mole(brenda_rogers)] is derived from modus ponens by 4 and [at(brenda_rogers, timestamp(2013,6,7,10,42,12.1), london)] by 6 and [at(spinnaker,timestamp(2013,6,7,10,42,12.1), rome)] by 13 and [~same_location(london,rome)] by 7
6. [at(brenda_rogers,timestamp(2013,6,7,10,42,12.1), london)] is derived from told([confirmed, source(agent2)], at(brenda_rogers, timestamp(2013,6,7,10,42,12.1), london))
7. [~same_location(london,rome)] is derived from told([geography], ~same_location(london,rome))

B. Reporting Analytical Results

The complexities of reasoning performed during Analysis can be abstracted and tailored to the user’s analysis style. A simple query interface built into Palantir complements the Palantir/Mephisto bridge described in section IV.B, allowing end-users to perform computationally intensive analysis while being able to remain ignorant of the formal logic. Pre-configured query templates within Palantir, shown in Fig. 5, are filled in by the user.

![Palantir Mephisto query interface](image-url)

Fig. 5. Palantir Mephisto query interface

Query results are reported using Palantir functionality. Links inferred by Mephisto are visualised on the Palantir Graph, pending validation by a HIL prior to committing that
link to the Palantir knowledgebase. Supporting evidence provided by the Why reasoner may aid the validation. A simple daughter-of relationship from Jessie to Graham fed into Mephisto shown in Fig. 6 forms part of the evidence for inferred influence Graham has over Jessie shown in Fig. 7.

Fig. 6. Daughter_of relationship visualised on the Palantir Graph

This HIL process lends itself well to low volumes of inferred data, allowing the human to manually accept or reject the machine's suggestions. Large volumes may be automatically accepted and added to the Palantir knowledgebase, allowing it to be explored along with other confirmed data.

Reporting analyses could also be performed in external analytical tools such as dedicated Social Network Analysis (SNA) tools. Users of such tools would greatly benefit from an ability to easily create and manage clean, inferred data from text, represented in a structure suitable for existing tools to run existing algorithmic analyses. Further work could be done to include these types of algorithmic analyses in the reasoning system.

To achieve higher level fusion a system needs to be able to assess situations and threats and present them to users. Virtual Advisors (VA) ([14], [15]) could form part of a story telling interface that could update the situation awareness of human users and be available for questioning. Such a VA could be external to Palantir, or embedded in Palantir and combined with machine-controlled navigation of Palantir to highlight the key information and provide an explanation of the situation being assessed. An automated storytelling interface is currently being implemented and is illustrated in Fig. 8.

Fig. 7. Inferred Influence relationship ready for HIL validation

Fig. 8. Inferencing with a virtual adviser interface

VII. CONCLUSION

The paper has outlined the MISFIT process for implementing soft fusion with a number of novel aspects: mixed initiative approach which identifies a human-machine division of labour; a process for ingesting heterogeneous data; semantic triage; semantic coding; automated inference; and the ability to present results through conventional and novel interfaces.

VIII. ACKNOWLEDGMENT

The authors would like to thank Dr Adam Saulwick, Dr Kerry Trentelman and Sean Bergin for their many contributions and insights during collaborative discussions that shaped this work. Additional thanks to Dr Poh Lian Choong, Dr Tim Pattison, Brenton Williams, Jennifer Biggs, Dr Glen Smith and Dr Richard Price.

REFERENCES