2023 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

FABULA: Intelligence Report Generation Using Retrieval-Augmented Narrative Construction

Priyanka Ranade
Department of CSEE
University of Maryland, Baltimore County
Baltimore, MD, USA
priyankaranade@umbc.edu

Anupam Joshi
Department of CSEE
University of Maryland, Baltimore County
Baltimore, MD, USA
joshi@umbc.edu

***Abstract*—Narrative construction is the process of representing disparate event information into a logical plot structure that models an end to end story. Intelligence analysis is an example of a domain that can benefit tremendously from narrative construction techniques, particularly in aiding analysts during the largely manual and costly process of synthesizing event information into comprehensive intelligence reports. Manual intelligence report generation is often prone to challenges such as integrating dynamic event information, writing fine-grained queries, and closing information gaps. This motivates the development of a system that retrieves and represents critical aspects of events in a form that aids in automatic generation of intelligence reports.**

We introduce a Retrieval Augmented Generation (RAG) approach to augment prompting of an autoregressive decoder by retrieving structured information asserted in a knowledge graph to generate targeted information based on a narrative plot model. We apply our approach to the problem of neural intelligence report generation and introduce FABULA, framework to augment intelligence analysis workflows using RAG. An analyst can use FABULA to query an Event Plot Graph (EPG) to retrieve relevant event plot points, which can be used to augment prompting of a Large Language Model (LLM) during intelligence report generation. Our evaluation studies show that the plot points included in the generated intelligence reports have high semantic relevance, high coherency, and low data redundancy.

***Index Terms*—retrieval augmented generation, large language models, knowledge graphs, narratives**

I. Introduction

The process by which information about critical events is disseminated, articulated, and shaped into news stories has greatly evolved since the proliferation of digital media and the World Wide Web. Intelligence analysts now take advantage of an abundance of online communication mediums to widely share and obtain reporting on a variety of critical events. Intelligence analysts rely heavily on manual techniques to extract evolving fine-grained event details over multiple Open Source Intelligence (OSINT) sources in *real-time*. This evaluated information is then manually documented within an intelligence report, which is used during tactical operations by decision makers to manage, evaluate, and keep updated on evolving events, such as breaking news occurrences.

Intelligence reports are inherently structured to communicate *narratives*, which are accounts of *interconnected event incidents and actors* (plot points) evolving through some notion of *time*. In journalism and storytelling, there have been several types of *narrative plot structures* proposed to organize and convey event information. One of the most well suited for intelligence analysis is the Inverted Plot Pyramid (IPP) narrative structure (Figure 2, Section II-A) which is designed specifically for conveying news event details. It is becoming more clear that the intelligence analysis domain can benefit tremendously from techniques in *computational narrative construction*, which utilize existing Information Retrieval (IR) methods such as document collection, query processing, and ranking to aid end users in comprehending disparate event information. Specifically, integrating narrative construction tasks to intelligence analysis workflows can alleviate the costly nature of intelligence report generation in three primary ways: (a) Automatically extracting event information from a dense collection of documents based on a schema, (b) Aiding in information triage during intelligence report generation, (c) Tracking and integrating evolving event information over time.

Recent advancements in Large Language Models (LLMs) have enabled state of the art results in automatic text generation tasks, presenting new opportunities in the computational narrative construction domain. For example, an end user can issue directed *prompts* to a generative LLM to automatically generate summaries that communicate end to end narratives about an event. Similarly, we envision that an *intelligence analyst* can prompt an LLM to automatically generate accurate intelligence reports about queried events. Despite these potential benefits, a direct application of LLMs for automatic intelligence report generation presents several limitations, such as: (a) Output hallucinations where the generated text contains non-factual, non-event related, and incomplete information, (b) Lack of provenance, attribution, and trust for knowledge sources used to generate responses. The AI community has started to address these challenges through a general approach called *Retrieval Augmented Generation* (RAG) which uses

ASONAM '23, November 6-9, 2023, Kusadasi, Turkey

ACM ISBN 979-8-4007-0409-3/23/11…$15.00
https://doi.org/10.1145/3625007.3627505

non-parametric memory to augment LLM generation.

Inspired by these recent developments, we develop FABULA, a framework that integrates a novel RAG approach for using narrative plot structures, LLMs, and knowledge graphs to automatically generate intelligence reports (Figure 1). The *main contributions* of this paper include:

- FABULA: A framework to augment intelligence analysis workflows. Analysts can use the system to automatically generate intelligence reports for events utilizing contextual narrative features found in OSINT (Section III).
- Retrieval-Augmented Generation (RAG) approach which retrieves plot points from a knowledge graph and provides them as input prompt sets for guided LLM intelligence report generation (Section V-C).

II. Related Work

In this section, we describe research on narratives, news event OSINT, knowledge representation, and data to text generation.

A. Narratives, Stories, & News

OSINT about *events*, are published via blogs, social networks, news sources. Events contain *plot points*, which are incidents that directly impact what happens next [1]. Events are communicated through the form of *narratives* which are accounts of interconnected plot points [2]. A seminal example of a narrative plot structure is the *Plot Pyramid Model* by Gustav Freytag, a five component framework that outlines thematic and temporal stages in generic storytelling [3]. The components are, *Introduction, Rising Action, Climax, Falling Action, and Denouement*. The plot points in the Freytag pyramid develop and conclude progressively over time, first leading to the development of the *climax* (introduction and rising action) and successively concluding to the *denouement* as a direct result of the climax (falling action). Unlike the pyramid, other narrative plot structures organize plot points in varying ways. For example, the *Fichtean Curve*, begins immediately with the rising action component, followed by a series of *crisis* (falling action) [4]. There are several more examples of other narrative plot structures such *The Hero's Journey* that model the development of events differently [4]. Our system FABULA, focuses specifically on modeling the plot structure of open source *news events*. A narrative plot structure specifically developed for communicating news stories is the *Inverted Plot Pyramid* (IPP) (Figure 2). IPP is a three component model that conveys the critical plot points in the first component *(Lead)*, the event developments *(Body)*, and the nonobligatory information at the end *(Tail)* [5].

B. Narrative Construction and Schemas

Sequencing disparate events from a variety of sources is known as *fragmented narrative construction* [2]. While the disparate nature of OSINT provides opportunities for users to obtain insights, it presents challenges for chaining accurate data across *noisy* sources [6]. There have been several methods that address this problem. One is by sequencing causal and temporal event shifts into *story chains*. Zhu et al. [7] defines a story chain as "a construction of news articles that reveal hidden relationships among different events". They utilize random walks on a bipartite graph to form a coherent story chain based on a query. Prior to this work, research projects have ordered news based on hierarchy [8], [9]. A novel method utilized in our FABULA system for extracting narratives is through the use of narrative schemas. Narrative schemas are models used to represent primary components of a narrative, such as actors, plot points, and actions [10].

C. Information Retrieval & Knowledge Representation

Information Retrieval (IR) systems have evolved from symbolic-based methods [11] to neural retrieval models [12]. These capture *semantic matches* using neural networks to build vectorized knowledge representations. Representation techniques such as Knowledge Graphs (KG), have been popularly employed to *support* IR tasks [13].

D. Knowledge Graph (KG) to Text Generation

The KG-to-text generation task is a form of semantic triple verbalization, which automatically generates descriptive text for a given KG [14], [15]. State of the art methods fine-tune text-to-text and generative decoder pre-trained models with KG-to-text datasets. One approach to this problem is *Retrieval Augmented Generation* (RAG), a method that aims to seed external data is a technique that uses retrieved data that is stored externally (like in a KG) from the foundation model, which is used to augment LLM prompting by injecting relevant retrieved information.

III. Methodology

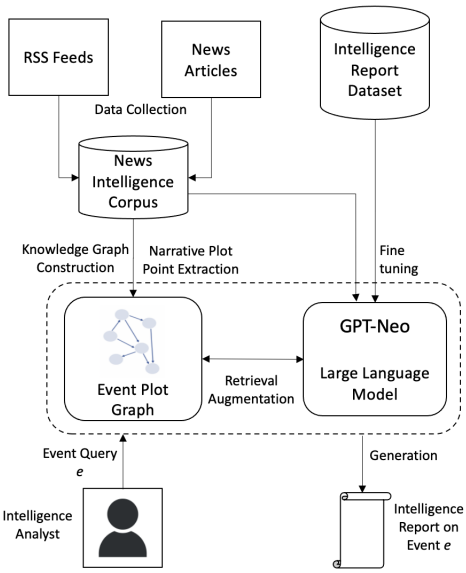

Fig. 1. FABULA System Architecture and Data Flow.

This work is guided by the following research question: *Does incorporating narrative-based features during Retrieval-Augmented Generation (RAG) produce intelligence reports with high semantic relevance, high event coherency, and little*

to no hallucination? Our approach is applied and evaluated specifically in the context of narrative construction for news events represented in open sourced news articles.

Suppose we have a set of randomly ordered articles d_1, d_2..., d_n, retrieved by a keyword search query about an event e, which contain several plot points p_1, p_2..., p_n. Each plot point is extracted, ranked, and ordered into a chronological sequence. Consider a real-world example where an *intelligence analyst* requires information about a critical event e. The analyst will input a query about e, retrieve relevant information from a multiple set of sources (news blogs, social media posts) to write a condensed intelligence report. We develop FABULA, an analyst-augmentation framework that integrates real-time news event retrieval, narrative schema-based information extraction and representation of event concept information, and retrieval-augmented generation (RAG) of intelligence reports.

Our approach is displayed in Figure 1. We begin by creating a *news intelligence corpus* from popular news sources and publicly available U.S Intelligence Community (IC) reports, described in Section IV-A. This corpus is further condensed through information extraction of plot points contained within news articles and follows the Inverted Plot Pyramid (IPP) narrative structure (Section IV-B). The extracted plot points are then asserted into an Event Plot Graph (EPG) using our base Event Narrative Ontology (ENO) schema (Section IV-C). Next, we fine-tune the LLM, GPT-Neo [16] using our news intelligence corpus and the extracted plot points V-A). FABULA's EPG is queried using Retrieval Augmented Generation (RAG) to control GPT-Neo's intelligence report generation process.

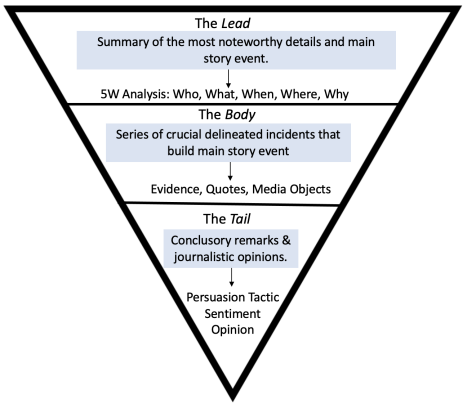

Fig. 2. The Inverted Pyramid Plot (IPP) model and associated text features.

IV. Event Intelligence Collection and Representation

In this section we describe methods for collecting open source news event data and organizing it in a knowledge representation based on the Inverted Pyramid Plot (IPP) news narrative model features.

A. News Intelligence Corpus

The first component of FABULA is a *news intelligence corpus* which contains streams of scraped public news articles, D and open source intelligence reports IR released by the U.S government. FABULA utilizes Really Simple Syndication (RSS) feed triggers for sources such as CNN [17], New York Times [18], CBS News [19], U.S. Department of State [20], and U.S. Department of Defense [21]. Each feed contains article metadata such as headline, author(s), abstract, and a web link. When updates are pushed by sources on their RSS feeds, FABULA utilizes the RSS web links to retrieve the corresponding webpages. We further extract information such as timestamps, news text, image links, video links from each article, using the BeautifulSoup web parser[1]. Each individual news article is represented in the collection of OSINT news set D.

$$D = \{d_1, d_2, ..., d_k, ..., d_n\}$$

Each d_k represents a news article and its headlines, author(s), timestamps, text, images, video links.

Set IR is composed of publicly available intelligence reports released by the United States Office of the Director of National Intelligence (ODNI) [2]. The ODNI intelligence reports (2005-2023) have a common structure that matches the Inverted Plot Pyramid narrative scheme (Section IV-B). Our final news intelligence corpus $D + IR$, contains to 3000 D news articles and 165 IR reports.

B. Event Plot Extraction

FABULA solves a *fragmented narrative construction* problem where given a set of relevant articles retrieved by a search query about event e, contain several plot points:

$$P = \{p_1, p_2, ..., p_i, ..., p_o\}$$

Where, p_i is a plot point that is extracted from D. The associated plot points for an event e, can be determined through information extraction based on classes in the *Inverted Plot Pyramid (IPP)* (Figure 2), a standardized narrative structure for communicating news events (Section II-A). The IPP narrative structure has three sub-categories: *Lead, Body,* and *Tail*. Definitions of the IPP sub-categories and associated sub-types are available in Table I. To extract the IPP-based plot points, we implement a *Narrative Plot Concept Extractor (NPCE)* that processes the set of news articles D. In the rest of this section, we further describe the methods used to extract the plot points described in Table I that form set P. Extraction occurs at each level of the IPP - *Lead, Body,* and *Tail*.

1) The Lead: Event information contained in the IPP Lead class describes the most noteworthy event details. As indicated in Table I, these details are expressed through the *5W communication device*: who? (p_{who}), what? (p_{what}), when? (p_{when}), where? (p_{where}), and why? (p_{why}). The answers to the 5W questions provide a circumstantial view of an event. The NPCE extracts the 5Ws using pre-trained Named Entity Recognition (NER) and Part of Speech (POS) Tagging models from the standardized spaCy NLP Framework [22]. The specific entity and relationship types extracted are provided in Table II. The spaCy NER was trained on the widely benchmarked

[1]https://www.crummy.com/software/BeautifulSoup/

[2]https://www.dni.gov/index.php/newsroom/reports-publications

Inverted Pyramid Class	Plot Element	Notation
Lead: Summary of the most noteworthy details and main story objective/event.	Who: Identification of the subject or persons involved.	p_{who}
	What: Occurences of scenes, incidents, artifacts, or actions.	p_{what}
	When: Recorded timestamps and dates.	p_{when}
	Where: Geographic regions and locations mentioned.	p_{where}
	Why: The cause and reason to describe event occurence.	p_{why}
Body: Series of crucial delineated incidents that build main story objective/event.	Evidence: Supporting details surrounding an event.	$p_{evidence}$
	Quotes: Phrases noted by involved persons.	p_{quote}
	Media (Photos): Digital Image Object (HTML DOM image element)	p_{photo}
	Media (Video): Recorded Video Object (HTML DOM video element)	p_{video}
	Media (Audio): Recorded Audio Object (HTML Dom audio Element.)	p_{audio}
Tail: Conclusory remarks and journalistic opinions.	Journalistic Opinion: Non-fact based judgements about event.	$p_{opinion}$
	Persuasion Tactic: Instances of rhetorical dimensions (ethos, pathos, logos)	p_{tactic}
	Sentiment: Emotional tone and affective state information.	$p_{sentiment}$

TABLE I
TEXTUAL FEATURES REPRESENTING PLOT ELEMENTS FOR EACH INVERTED PYRAMID PLOT MODEL CLASS.

Lead Class Attributes	OntoNotes Entity and Relationship Types
Who	PERSON, NORP, ORG
What	EVENT, FAC, PRODUCT, WORK_OF_ART, LAW, MONEY, LANGUAGE, PERCENT, QUANTITY, ORDINAL, CARDINAL
When	DATE, TIME
Where	GPE, LOC
Why	cause, causing, caused by, because, since, after, for, as and of

TABLE II
ENTITY AND RELATIONSHIP TYPES FOR THE 5W CLASSES

OntoNotes dataset. [23]. The *why* category in particular, is extracted using the spaCy Part of Speech (POS) tagger, which locates sentences containing causal relationships (prepositions, verbs, conjunctions) between entities.

2) The Body: The most significant plot points, involving major incidents and themes of an event are communicated in the body of an article. The incidents are typically written as factual occurrences to form a delineated sequence of information that builds the main story objective and overall situational awareness of an event. FABULA's NPCE utilizes a FAISS-based clustering [24] and regex approaches to extract Body category event information (Table I), which include evidences ($p_{evidence}$), quotes (p_{quote}), and media objects stored as URLs (p_{photo}, p_{video}, p_{audio}).

3) The Tail: The Tail category contains conclusive remarks, journalistic opinions ($p_{opinion}$), persuasion tactics (p_{tactic}), and perceived sentiment ($p_{sentiment}$) of the article author and source organization. These features do not contain plot points that impact the *development* of an event, but rather can be used by an analyst to understand factors that may influence the author's narrative.

Opinion ($p_{opinion}$) and persuasion tactic (p_{tactic}) identification is a multi-label task at the paragraph level. We utilize a gold standard model developed for the ACL Semantic Evaluation Task [25]. This was trained on the only publicly available human-labeled corpus specifically developed for persuasion language extraction [25]. We treat each article in our set D as a holdout sample and use the provided model to extract the persuasion tactics and the corresponding text sample. For sentiment detection, we utilize the spaCy sentiment analysis program *polarity* to extract positive, negative, and neutral sentiment for each article.

The NPCE output of the 3-level *Lead, Body,* and *Tail* extraction populates the event plot points set P. We maintain the mapping between the extracted set P and the associated documents in set D for LLM prompt tuning, described further in Section V-C. In the next subsection, we describe methods we use to assert set P into FABULA's Event Plot Graph (EPG).

C. Event Narrative Ontology & Event Plot Graph

We introduce the *Event Narrative Ontology* (ENO), a Web Ontology Language (OWL)-based knowledge representation. ENO allows FABULA to store event information based on extracted plot point category features, described in the previous section. ENO serves as the base schema for FABULA's *Event Plot Graph* (EPG). NPCE's output, set P (Section IV-B) is asserted in the EPG using ENO. ENO classes and properties have been constructed using the elements of the *Inverted Plot Pyramid (IPP)* narrative scheme (Figure 2, Section II-A). While we incorporate the IPP scheme due to its relevance to intelligence analysis in particular, variants of ENO can be built based on a variety of narrative theories.

The EPG contains a set of stored as Resource Description Framework (RDF) triples donated as G,

$$G = \{(s, p, o) | s, o \epsilon I, p \epsilon R\} \tag{1}$$

where, I and R denote the instances and relations stored in G. (s, p, o) is a single triple in G and denotes the relation p between two entities s and o. Below is a description of ENO classes and properties.

Classes in ENO: ENO contains a total of 16 classes and subclasses: two generic classes: $NewsArticle$, $PlotPoint$, and 14 subclasses, all of which are of type $owl:Class$. The classes organize the information extracted in Section IV-B in a form that incorporates the IPP narrative scheme. Descriptions are as follows:

- `NewsArticle`: Instances contain identifiers for news articles and metadata such as publisher, author, URL, etc.
- `PlotPoint`: Describes IPP narrative elements (Section IV-B). It has following subclasses:
  1) `Lead:` : Plot points that include noteworthy details. Subclasses are categorized below:
  - `Who`: Person, affiliation, organization (p_{who}).

- `What`: Incidents, artifacts, or actions (p_{what}).
- `When`: Recorded timestamps and dates (p_{when}).
- `Where`: Geographic locations/regions (p_{where}).
- `Why`: Event causal descriptions (p_{why}).

2) `Body`: : Plot points that describe news article objective. It has the following subclasses:

- `Evidence` Supporting details ($p_{evidence}$).
- `Quote`: Text demarcated by quotation marks (p_{quote}).
- `MediaObject`: Class representing photo (p_{photo}), audio (p_{audio}), and video (p_{video}) DOM objects.

3) `Tail`: : Plot points representing closing remarks and opinions. It has the following subclasses:

- `Opinion`: Extracted author opinion ($p_{opinion}$).
- `PersTactic`: Persuasion technique (p_{tactic}).
- `Sentiment`: Tone, emotion, mood ($p_{sentiment}$).

Properties in ENO: To encode extracted relations ENO incorporates multiple object and data properties that can be asserted in the EPG. We describe some of these below:

- $articleHeadline$: Extracted string literal from *NewsArticle* instance denoting the article headline.
- $authorOfArticle$: Extracted string literal from *NewsArticle* instance denoting author(s) of the article.
- $publishedBy$: Data property denoting the publishing source of the *NewsArticle* instance.
- $publishedDate$: Timestamp for $NewsArticle$ instance.
- $hasPlotPoint$: This object property helps codify relations between instances of the extracted IPP $PlotPoint$ and its associated $NewsArticle$. The property can be inherited by instances of $PlotPoint$ subclasses.

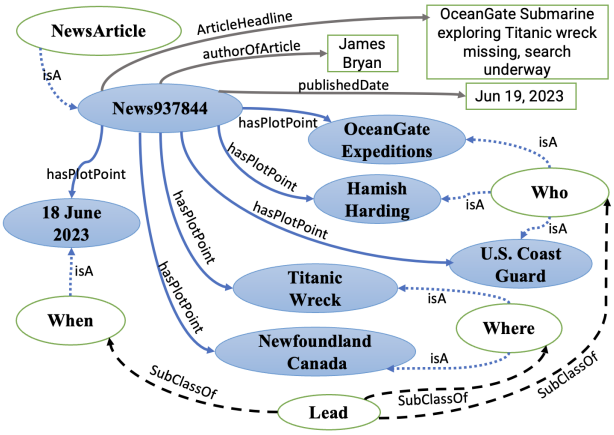

Fig. 3. Populated EPG Sub-graph for the 2023 Titan Submersible Implosion.

For example, Figure 3, shows an EPG sub-graph about the *2023 Titan Submersible Implosion* event. The graph represents one news article *News937844*. This sub-graph displays the *Lead* narrative plot points about a *catastrophic implosion* (p_{what}), involving the *U.S Coast Guard*, *OceanGate Expeditions*, and *Hamish Harding* (p_{who}), occurring at the *Titanic Wreck* site, near *Newfoundland, Canada* (p_{where}).

Though not shown in this subgraph, we were able to extract *Body* and *Tail* narrative plot points, which were asserted in the EPG. These include *Body* instances: four $p_{evidence}$ pieces, two p_{quote} and one p_{image}, and *Tail* instances: an *attack on reputation* p_{tactic} and *negative* $p_{sentiment}$ utilized by the article author *James Bryan*. The EPG G constructed from the news stream collection D is next utilized in automatic intelligence report generation.

V. Retrieval-Augmented Intelligence Report Generation

An intelligence analyst can use FABULA to delineate event plot elements stored in the EPG G, for automatic generation of intelligence report Y about event e. FABULA implements a Retrieval-Augmented Generation (RAG) approach that queries the EPG to retrieve a *narrative prompt set* about event e (Figure V-A). The set serves as a prompt to a Large Language Model (LLM) during report generation. There are three primary steps to our approach: (1) Fine-Tuning an LLM with $D + IR$, (2) Data-to-text prefix-tuning, and (3) SPARQL template LLM prompting for report generation.

Let V denote the vocabulary set of the report generation task. The desired *target output* is to generate report text denoted by Y utilizing the LLM where,

$$Y = (w_1, w_2, ..., w_j, ..., w_T)$$

$w_j \epsilon V$ is a single word in the generated report Y. To generate Y, we first fine-tune the LLM GPT-Neo [16] with our news intelligence corpus $D + IR$. Fine-tuning GPT-Neo using $D + IR$ has two advantages. First, it augments the existing vocabulary of GPT-Neo ($V_{GPT-Neo}$) with the vocabulary of the news intelligence corpus (V_{D+IR}) which is equivalent to the vocabulary of FABULA's EPG. After the fine-tuning report generation task, vocabulary V includes both $V_{GPT-Neo}$ and V_{D+IR}. Second, fine-tuning with public intelligence reports IR provides the LLM with examples of the desired document structure for output Y.

To generate the intelligence report Y, GPT-Neo takes as input a *narrative prompt set*, i.e. a set of narrative plot points about the event e stored in the subgraph $G' \epsilon G$.

$$G' = \{(s_e, p, o_e) | s_e, o_e \epsilon I_e, p \epsilon R\}$$

where, $I_e \epsilon I$ and R denote the instances relevant to query e and relations stored in G (Equation (1). The determination of the narrative plot points in G' retrieved from G, is implemented using the SPARQL Protocol and RDF Query Language templates executed on FABULA's EPG G.

The narrative prompt set serves as input to the fine-tuned GPT-Neo LLM that outputs the intelligence report Y. Each component of our approach is further described in the rest of this section.

A. Fine-Tuning GPT-Neo

Fine-tuning is an example of transfer learning, a method that seeds additional domain knowledge to a pre-trained Large Language Model, without training all parameters from scratch. We fine-tune the 1.3B parameter GPT-Neo decoder [16]. The original GPT-Neo model was trained with the *Pile* dataset [26], which is an 800GB English text corpus that consists of

22 high quality datasets. Fine-tuning GPT-Neo using $D + IR$ (Section IV-A) augments the existing vocabulary of GPT-Neo ($V_{GPT-Neo}$) with the vocabulary of the news intelligence corpus (V_{D+IR}) and allows GPT-Neo to model the format and syntactic style of known intelligence reporting, such as that available in set IR, which closely follows the IPP narrative structure.

During fine-tuning, we divide the training set in a 35% train and test split. We use batch size 16 and learning rate 0.0001, trained for 12 hours. The output of the model is a conditional probability of each word in the target text given the input and the previously generated words. We report a perplexity value (the exponential of the cross-entropy loss) of 11.14.

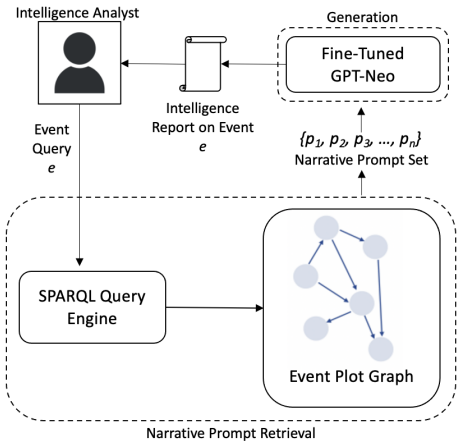

Fig. 4. FABULA's Retrieval-Augmented Generation (RAG) of Intelligence Report about event e.

B. Prefix-tuning with Narrative Prompt Sets

Traditionally, an autoregressive decoder like GPT-Neo requires a sentence based prompt to initiate generation. To modify this requirement and to enable prompting using a narrative prompt set for intelligence report Y generation, we require data-to-text *prefix-tuning* [27].

Prefix-tuning is a lightweight supplement to the fine-tuned GPT-Neo. This method keeps the GPT-Neo model parameters frozen and optimizes a sequence of continuous task-specific *virtual vectors* to the key and value matrices. When the tuning process is complete, the virtual tokens are stored in a lookup table, used during inference. We use the article to plot point mappings generated by the NPCE, described in Section IV-B and the HuggingFace Parameter-Efficient Fine-Tuning (PEFT) module for prefix-tuning. We use a beam decoding scheme and observed that adding more keywords provides increased supervision to the model and narrows the distribution of keyword context in the entire training dataset, leading to more accurate generation. More information on our evaluation can be found in Section VI.

C. Large Language Model Prompting using FABULA's EPG

A FABULA generated intelligence report Y on event e should contain reliable and consistent information. Y should only contain plot points that are relevant to the input query event e, and should exclude non-event related details. Achieving this criteria is not plausible by solely utilizing non-deterministic generative LLMs such as GPT-Neo, which are prone to *output hallucinations* [28]. We combat hallucinations and fulfill the above criteria by using a RAG-based approach by retrieving event narrative plot points stored in the EPG (Section IV-B) to control the prefix-tuned LLM generated output report. Our RAG approach is displayed in Figure V-A.

To create a narrative prompt set for the intelligence analyst's event query e, FABULA utilizes SPARQL Protocol and RDF Query Language (SPARQL) templates. These queries can be executed on the EPG to retrieve a set of plot points related to event query e. For example, the SPARQL query to output the *Lead* narrative plot points for the event query $e =$ "*Oceangate*" has been shown in Listing 1. FABULA includes a set of SPARQL query templates that can be leveraged to build a *narrative prompt set* for the intelligence analyst query e. This prompt set serves as the input to the prefix-tuned GPT-Neo that outputs the intelligence report Y. In the next section, we describe evaluation of the generated intelligence reports. Excerpts from an intelligence report generated by FABULA are available in Table V-C.

```
SELECT Distinct ?x Where {
WHERE {
   ?x rdf:type narr:Who. ?x rdf:type narr:What.
   ?x rdf:type narr:When. ?x rdf:type narr:Where.
   ?x rdf:type narr:Why. ?y narr:hasPlotPoint ?x.
   ?y rdf:type narr:NewsArticle.
   ?z narr:ArticleHeadline ?y.
   FILTER regex(str(?z), "Oceangate").}
Output:
<OceanGate Expeditions, Stockton Rush, Paul-Henri
Nargeloe, Hamish Harding, Shahzada Dawood, Suleman
Dawood, Titanic, wreck, submersible, 18 June,
370 miles, Newfoundland, Canada, Atlantic Ocean>
```

Listing 1. SPARQL query to retrieve Lead plot points for $e =$ "*Oceangate*" and corresponding output

VI. Experimentation & Evaluation

Our evaluation study is composed of two experimental approaches. First, we automatically evaluate the semantic and syntactic quality of the generated reports using Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [29]. Second, we qualitatively evaluate the reports through a human evaluation done by a group of 3 analysts.

Table V-C, provides samples of the generated intelligence reports for the *2023 Oceangate Submersible Disaster* and the *2023 Ohio Train Derailment Disaster* events. Column one displays the narrative prompt templates that were used for EPG plot point instance extraction (See Section V-C). Each IPP class has an associated derived narrative prompt set (Column 2), which is converted into a linearized prompt of keywords, used as input to GPT-Neo for guided text generation (See Section V-C). The bolded text in the generated samples (Table V-C, Column 3) represents occurrences of the prompt

2023 Oceangate Submersible Disaster		
Prompt Template	**Prompt**	**Generated Text**
Lead Template <p_{who}, p_{what}, p_{when}, p_{where},>	<OceanGate Expeditions, Stockton Rush, Paul-Henri Nargeloe, Hamish Harding, Shahzada Dawood, Suleman Dawood, Titanic, wreck, site, submersible, 18 June, 370 miles, Newfoundland, Canada, 1 hour and 45 minutes, Atlantic Ocean>	The **OceanGate Expeditions** submarine is the subject of the rescue operation for **Hamish Harding**, **Stockton Rush**, **Paul-Henri Nargeloe**, **Shahzada Dawood**, **Suleman Dawood** were on board for an expedition into the **Titanic wreck site**. The Titan **submersible** went missing **18 June** about **370 miles** off the coast of **Newfoundland** in **Canada**. The submersible lost contact just **1 hour and 45 minutes** after diving into the **Atlantic Ocean**.
Body Template <$p_{evidence}$, p_{quote},>	<debris, ROV, seafloor, titanic, sonar, banging, noises, implosion, injury, dead, safety, regulations, failure, United States Coast Guard said "likely died from the failure of the craft">	**Debris** was found by a **ROV** near the **Titanic** wreck. **Sonar** detects **banging** sounds from underneath the water. The banging sounds indicate an unfortunate **implosion**. **The United States Coast Guard** claims the crew and passengers died from the implosion.
Tail Template <p_{tactic},>	<overshadowed the bigger Greece migrant vessel disaster>	Coverage **overshadowed** the greater Mediterranean disaster.
2023 Ohio Train Derailment Disaster		
Prompt Template	**Prompt**	**Generated Text**
Lead Template <p_{who}, p_{what}, p_{when}, p_{where},>	<Norfolk Southern Train, chemicals, East Palestine, Ohio, 9pm, 3 February>	**Norfolk Southern Train** carrying hazardous **chemicals** suffered a mechanical failure. The derailment took place in the village of **East Palestine, Ohio** at **9pm** on **3 February**.
Body Template <$p_{evidence}$, p_{quote},>	<derailment, cars, EPA, hazardous, chemicals, contaminants, cancer, gas, safety, The EPA said it "did not detect chemical contaminants at concerning levels in the hours after venting.">	The cars had **derailed**, including the cars that were carrying **hazardous** materials. The **EPA** claims no concerning health risks. The chemicals are linked to increased risk of **cancer**.
Tail Template <p_{tactic},>	<overblown characterisations about the derailment disaster>	Reporters made **overblown** accusations about the derailment disaster.

TABLE III
GUIDED INTELLIGENCE REPORT GENERATION USING IPP PROMPT TEMPLATES.

keywords present in the final generated intelligence report. We limit the generation to 500 words to avoid model hallucination and inclusion of non-event related information. We found prompting with longer sequences of keywords (such as those extracted from the Lead and Body templates) resulted in more tightly coherent and semantically relevant generations, versus the Tail template, which mostly only included the extracted persuasion tactic as a prompt. We found that for instances such as these, the model would deflect from the event and sometimes include non-relevant information. Therefore, for tail generation, we limit the output to 100 words.

A. Quantitative Evaluation

We utilize *Rouge scores* to quantitatively evaluate the efficacy of our system in generating syntactically accurate intelligence reports [29]. Rouge-n in particular, allows us to compute the ratios of overlapping n-grams between generated reports and reference text. In particular, we use event descriptions extracted from Wikipedia as a reference set to calculate the syntactic overlap between these event descriptions and FABULA's generated intelligence reports. The Wikipedia event descriptions are derived based on content from a variety of online sources and reporters, written by human volunteers and overseen by Wikipedia moderators. These Wikipedia reference descriptions help provide us with a lateral publicly available comparison for FABULA's fragmented narrative construction. We use the Wikipedia Python library summaries endpoint [30] to extract event descriptors for 50 different public events we randomly selected from $D + IR$ and calculate Rouge-1 and Rouge-2. This provides term-based measures to quantify topic-level semantic relevance and syntactic quality [29]. Our results are displayed in Table VI-B. Rouge-1 refers to overlap of unigrams between FABULA's reports and Wikipedia's event descriptors while Rouge-2 refers to the overlap of bigrams.

B. Human Evaluation (Qualitative) Study

After evaluating the general efficacy of our model using quantitative metrics, we also conduct a human evaluation study to validate FABULA's capability required specifically for the *intelligence report generation* task. Given the high cost of this evaluation, we task a group of 3 analysts to score reports across 5 randomly selected events with two aspects: *factual correctness* and *language fluency*. The first criterion evaluates how well the generated report conveyed the overall narrative of the event. The second criterion evaluates grammatically correctness and fluency of the generated intelligence report.

The analysts were given a set of 5 articles per event (total 5 events), and were tasked to manually create a *single* report to convey the critical aspects across the set of 5 articles, for each separate event. This helped the analysts understand the narrative details for each of the 5 events. We then tasked the analysts to recommend IPP plot points from each of the 5 events. We compute Cohen's kappa [31] to measure inter-annotator agreement for each of the recommended IPP plot points, keeping only the plot points that scored higher than 0.6. This helped us derive a gold standard set of 78 plot points that analysts want in the generated report, referred to as set $Gold$. We then identify the number of IPP plot points in the FABULA's generated reports for the 5 events. The number of IPP plot points in the generated reports overlapping with $Gold$ is called support (denoted as #Supp). The number of missing plot points that were in $Gold$ and not present in FABULA's

generated reports are called contradicting plot points (denoted as #Cont). The average scores are displayed in Table VI-B. These are computed against the $Gold$ average of 15.6.

Quantitative Results		
Rouge-1	**Rouge-**2	
61.27	24.51	
Qualitative Results		
#Supp	**#Cont**	**Fluency**
15.6	13.2	4.2

TABLE IV
QUANTITATIVE QUALITATIVE RESULTS FOR GENERATED INTELLIGENCE REPORTS.

To score grammatical correctness and linguistic *fluency*, we adopt a 5-point Likert scale [32] ranging from 1-point ("Unacceptable") to 5-point ("Very Acceptable") tasking analysts to score the 5 FABULA generated reports. The *Fluency* column in Table VI-B reports averaged score (4.2/5) from 5 human analysts over the 5 generated reports.

VII. CONCLUSION & FUTURE WORK

Deriving narratives about events using disparately sourced information is a challenging task for an intelligence analyst. Analysts heavily rely on traditional, *manual* techniques to parse large amounts of noisy OSINT data to create cohesive intelligence reports. These manual methods do not provide complete situational awareness and are prone to information gaps and inaccurate representations of dynamic events.In this paper, we have described our framework FABULA (Figure 1), that integrates real-time news event retrieval, narrative schema-based information extraction and representation of event concept information, and retrieval-augmented generation (RAG) of intelligence reports.

We evaluate the generated reports using quantitative Rouge evaluation metrics and through a qualitative human evaluation study. Our results show that the plot points constructed within the generated intelligence report have high semantic relevance, high coherency, and low data redundancy. In planned future work, we are exploring methods to train transformer based language models to automatically learn the structure of a variety of plot models. It is a non-uniform process to identify narrative features in natural language. We are pursuing strategies to transfer the classified plot relationships to broader events and domains.

REFERENCES

[1] Wendy G Lehnert. Plot units and narrative summarization. *Cognitive science*, 5(4):293–331, 1981.

[2] Priyanka Ranade, Sanorita Dey, Anupam Joshi, and Tim Finin. Computational understanding of narratives: A survey. *IEEE Access*, 2022.

[3] Ryan L Boyd, Kate G Blackburn, and James W Pennebaker. The narrative arc: Revealing core narrative structures through text analysis. *Science advances*, 6(32):eaba2196, 2020.

[4] Kristin Thompson. *Storytelling in the new Hollywood: Understanding classical narrative technique*. Harvard University Press, 1999.

[5] Brian Keith, Michael Horning, and Tanushree Mitra. Evaluating the inverted pyramid structure through automatic 5w1h extraction and summarization. *Computational Journalism C+ J*, 2020.

[6] Faten El Outa, Matteo Francia, Patrick Marcel, Veronika Peralta, and Panos Vassiliadis. Supporting the generation of data narratives. In *ER Forum/Posters/Demos*, pages 168–172, 2020.

[7] Xianshu Zhu and Tim Oates. Finding story chains in newswire articles. In *2012 IEEE 13th International Conference on Information Reuse & Integration (IRI)*, pages 93–100. IEEE, 2012.

[8] Dafna Shahaf and Carlos Guestrin. Connecting the dots between news articles. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010.

[9] Gabriel Pui Cheong Fung, Jeffrey Xu Yu, Huan Liu, and Philip S Yu. Time-dependent event hierarchy construction. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 300–309, 2007.

[10] Bartalesi Valentina Meghini, Carlo and Daniele Metilli. Representing narratives in digital libraries: The narrative ontology. *Semantic Web*.

[11] Amit Singhal, Chris Buckley, and Manclar Mitra. Pivoted document length normalization. In *Acm sigir forum*, volume 51, pages 176–184. ACM New York, NY, USA, 2017.

[12] Nick Craswell, W Bruce Croft, Maarten de Rijke, Jiafeng Guo, and Bhaskar Mitra. Neural information retrieval: introduction to the special issue. *Information Retrieval Journal*, 21:107–110, 2018.

[13] Laura Dietz, Alexander Kotov, and Edgar Meij. Utilizing knowledge graphs for text-centric information retrieval. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 1387–1390, 2018.

[14] Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. Text generation from knowledge graphs with graph transformers. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics:*, 2019.

[15] Pei Ke, Haozhe Ji, Yu Ran, Xin Cui, Liwei Wang, Linfeng Song, Xiaoyan Zhu, and Minlie Huang. Jointgt: Graph-text joint representation learning for text generation from knowledge graphs. *arXiv preprint arXiv:2106.10502*, 2021.

[16] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow. 2021.

[17] CNN. https://www.cnn.com/services/rss/.

[18] NYTimes. https://www.nytimes.com/rss/.

[19] CBS New. https://www.cbsnews.com/rss/.

[20] U.S. Department of State. https://www.state.gov/rss-feeds/.

[21] U.S. Department of Defense. https://www.defense.gov/news/rss/.

[22] spacy nlp framework. https://spacy.io/, version = , date = 2023,.

[23] Xavier Schmitt, Sylvain Kubler, Jérémy Robert, Mike Papadakis, and Yves LeTraon. A replicable comparison study of ner software: Stanfordnlp, nltk, opennlp, spacy, gate. In *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, pages 338–343. IEEE, 2019.

[24] Hervé Jégou, Matthijs Douze, Jeff Johnson, Lucas Hosseini, and Chengqi Deng. Faiss: Similarity search and clustering of dense vectors library. *Astrophysics Source Code Library*, pages ascl–2210, 2022.

[25] Jakub Piskorski, Nicolas Stefanovitch, Giovanni Da San Martino, and Preslav Nakov. Semeval-2023 task 3: Detecting the category, the framing, and the persuasion techniques in online news in a multilingual setup. In *Proceedings of the The 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 2343–2361, 2023.

[26] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

[27] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[28] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*, 2021.

[29] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[30] Wikipedia. Wikipedia python library. https://pypi.org/project/wikipedia/.

[31] Mary L McHugh. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282, 2012.

[32] I Elaine Allen and Christopher A Seaman. Likert scales and data analyses. *Quality progress*, 40(7):64–65, 2007.