ScriptPulse: A system designed for content creators' script planning

Ruichen Zheng
UCLA MSCS
(USA)
ruichen.zheng118@gmail.com

Weihan Qu
UCLA MSCS
(USA)
weihanqu@g.ucla.edu

Zihan Jiang
UCLA MSCS
(USA)
joannajiang321@g.ucla.edu

ACM Reference Format:
Ruichen Zheng, Weihan Qu, and Zihan Jiang. 2018. ScriptPulse: A system designed for content creators' script planning. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 11 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Recent trends indicate a decreasing attention span, especially among Gen Z audiences, attributed largely to the growing popularity of short-form videos. This shift poses significant challenges not only for viewers but especially for content creators, who struggle to keep viewers engaged and maintain consistent retention rates in their content. Initial user research with both audiences and creators highlighted that the challenges of viewer engagement often arise from the creators' inability to clearly assess and optimize content structure and pacing effectively before production. Through surveys and interviews, content creators expressed that existing methods largely rely on intuition and trial-and-error, lacking structured guidance and actionable insights to improve audience engagement systematically.

Problem Statement: Content creators struggle to optimize videos for audience engagement and retention, as reading through scripts alone makes it hard to assess structure, key points, and flow. Existing tools offer basic analytics but lack actionable insights on how script structure impacts engagement. There is a need for a tool that visually maps script structure while providing actionable feedback, helping creators refine their content efficiently before production.

To address this gap, we propose a novel system, *ScriptPulse*, designed to help content creators visually assess and iteratively refine their video scripts. By providing an intuitive visualization of script composition alongside detailed, AI-generated feedback, our system enables creators to optimize content structure proactively, resulting in improved audience retention and a streamlined creative workflow.

Conference acronym 'XX, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/2018/06
https://doi.org/XXXXXXX.XXXXXXX

2 Related Work

2.1 Script Analysis and Audience Engagement (Academic Research)

Research in computational storytelling and video analytics has explored techniques for analyzing narrative structure, pacing, and audience engagement. For example, Chiu et al. developed a method to evaluate screenplay quality using linguistic cues and narratology-inspired features, treating award nominations as a proxy for script success [1]. Their approach identifies classic storytelling elements (e.g., three-act structure with key plot points) to predict critical acclaim. Others have proposed detecting narrative turning points to analyze story flow and structure [2], aligning with theories that well-paced sequences of setup, conflict, and resolution influence a script's impact. Beyond films, audience engagement modeling has been studied in interactive reading platforms: Fong and Gui introduced a framework leveraging Large Language Models (LLMs) to simulate readers' expectations and surprise, which improves predictions of which story chapters keep audiences hooked [3]. In the video domain, empirical studies of online content reveal how production and pacing affect engagement [4]. These works illustrate the value of analyzing script content and timing to anticipate audience reactions.

ScriptPulse builds on this foundation by providing creators with automated script analysis (e.g., highlighting structural beats or emotional arcs) and visualizing pacing metrics, bridging academic insights on narrative quality and engagement into a practical tool for content creators. Unlike prior studies that focus on offline prediction models or post-hoc analysis, ScriptPulse offers real-time feedback during the writing process, helping creators iteratively refine their scripts for optimal storytelling and viewer retention.

2.2 LLM-Powered Writing Assistance

The advent of large language models has spurred a wave of AI-assisted writing tools in both research and practice. Systems like *Wordcraft* demonstrate how GPT-3 can act as

a co-writer in creative fiction, with an interactive editor that generates story continuations or rewrites in response to user prompts [5]. DeepMind's *Dramatron* uses hierarchical prompt-chaining with LLMs to help screenwriters generate coherent scripts complete with characters, story beats, and dialogues [6]. These systems show that LLMs can contribute ideas, alternative phrasings, and even structured content for narrative creation. General-purpose chatbots (e.g., ChatGPT) and commercial writing AIs like *Jasper* also assist writers by suggesting text or rephrasing content in various tones [7]. Such LLM-based tools excel at generating and transforming text, but their feedback is often unstructured and not tailored to specific storytelling analytics.

ScriptPulse differs by using LLMs not just to generate text, but to provide targeted feedback on a video script's qualities—for example, flagging inconsistent tone, suggesting pacing adjustments, or summarizing scene intensity. While prior LLM tools function as co-creators, ScriptPulse acts more as an intelligent analyst, leveraging language models to critique and visualize aspects of a script's narrative dynamics. This approach builds on the success of LLMs in creative support but focuses on diagnosing and enhancing story structure and engagement factors, rather than writing the script from scratch.

3 User Research

Our initial research began with the broad topic of attention span, but we soon realized that it includes many factors beyond our control. To narrow our focus, we examined what aspects could be actively influenced—leading us to consider the role of content quality in audience engagement. While decreased attention spans are often cited as a reason why people struggle to watch long videos or get distracted, our research suggested that disengagement is not solely due to attention span limitations, but also to the content itself not being engaging enough. To explore this further, we conducted user research on both audiences and content creators, including surveys for both groups and interviews with content creators. This helped us understand not only what influences audience retention but also the challenges creators face in keeping viewers engaged. Ultimately, these insights led us to shift our focus from attention span itself to how content creators can structure and present their videos more effectively to retain audience interest. By centering our research on content creators, we aimed to provide actionable insights that could help them optimize their videos for better engagement.

3.1 Survey

Assumptions and Setup: We deployed a survey to invest general public's view on short vs. long videos and their personal experiences with and changes of attention span. Before conducting the survey, we have made a few assumptions about the user and usage context of our study, including both short video audiences and content creators:

- **Language and Age:** Primarily younger users (17 35 yo), English speaking or bilingual (English/Chinese)
- **Context:** Short videos are often consumed during fragmented time slots, such as commuting or breaks, usually watched alone. Long videos are more commonly watched during longer time slots, like during lunch or dinner, possibly with friends or family.
- **User Traits:** Users are generally comfortable with digital platforms and capable of multitasking.
- **Device:** Short videos are predominantly watched on mobile devices, whereas longer videos may also be viewed on laptops or tablets.
- **Viewing Context:**
  - Short videos (<30 seconds) are typically consumed alone during fragmented times such as commuting or short breaks.
  - Long videos (>30 seconds) are more likely watched during extended periods (e.g., meals), potentially with friends or family.

We formulated the following open-ended research questions to validate assumptions and explore user behaviors:

- Does frequent consumption of short videos truly correlate with a decrease in user attention span?
- What are the underlying reasons users prefer short videos? Is it due to inherently shorter attention spans, specific content characteristics, or contextual factors?

To address these questions, we designed and conducted our survey, targeting general video audiences to quantify their viewing habits, preferences, and perceived attention span changes. Surveys allowed us to collect large-scale quantitative data efficiently, revealing general trends about attention span concerns, video engagement factors, and viewing motivations. We deployed a Google Form survey primarily through UCLA student networks and WeChat groups due to time constraints. Thus, most respondents were Chinese-speaking students.

Survey Results: In total we collected 27 responses. Our survey provided valuable insights into user viewing habits, attention span perceptions, and motivations. First, regarding the question of attention span declines:

- **Attention Decline:** According to Figure 1, A significant majority (81.5%) of respondents reported experiencing a decline in attention span, particularly noticeable in academic and task-oriented activities such as reading, classroom focus, and time management, as presented in 2.
- **Short Video Consumption:** Despite varied consumption levels of short videos (ranging from 25% to 75% daily viewing), no clear correlation emerged between frequency of short video viewing and reported attention difficulties.

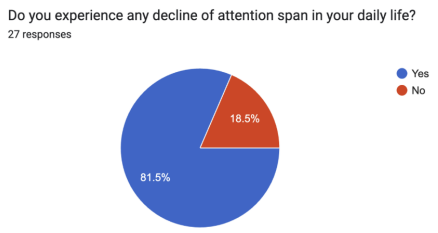

Figure 1. Do Respondents Experience A Decline In Attention Span?

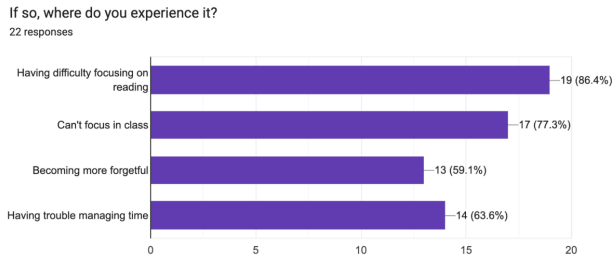

Figure 2. How do Respondents Experience such decline?

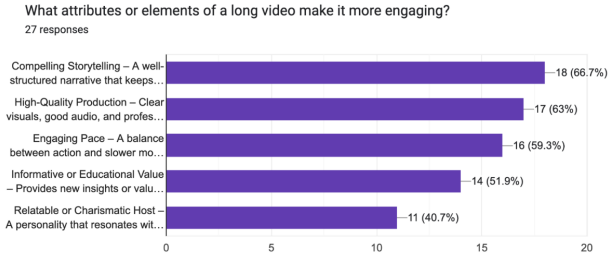

Figure 3. What makes a long video engaging?

- **Perceived Causes:** Opinions on whether short video viewing directly contributes to reduced attention span were evenly split—53.8% of respondents attributed their decreased attention to short videos, yet no direct correlation was found between their opinion and actual viewing frequency.
- **Engagement Factors:** As presented in Figure 3, respondents indicated the most effective factors for maintaining attention in videos were compelling storytelling, high production quality, and engaging pacing.

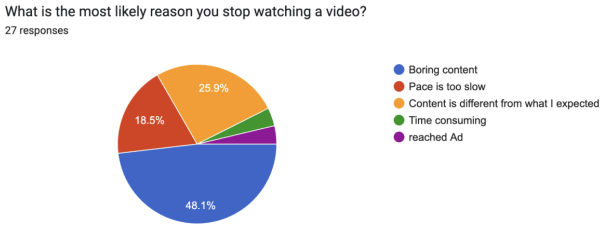

Figure 4. Why do people stop watching a video?

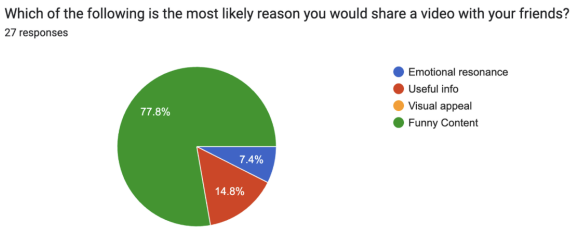

Figure 5. Why do people want to share a video?

Interestingly, approximately **30% of participants** admitted they struggled to maintain attention even while completing this survey, highlighting the significance of attention-related challenges.

Then, regarding user engagement and sharing behavior:

- **Reasons for Disengagement:** As shown in Figure 4, the primary reasons respondents stopped watching videos were:
  - Content perceived as boring or uninteresting (48.1%)
  - Content differing from viewer expectations (25.9%)
  - Slow pacing (18.5%)
- **Motivation for Sharing:** Respondents overwhelmingly indicated that humorous content (77.8%) was the primary motivator for sharing videos (Figure 5). This strongly suggests that content evoking strong positive emotions, particularly joy, significantly increases shareability.

These insights clearly demonstrate the importance of targeted, engaging content, and emphasize humor and storytelling as critical elements in effective video production.

3.2 Interview

To better understand the challenges content creators face and how they approach audience engagement, we conducted interviews with five content creators from different backgrounds. Their experiences revealed common trends and key insights about content formats, platform strategies, audience

engagement tactics, and perceptions of attention span.

1. Content Goals and Platform Strategies
   - Content creators have different motivations, ranging from self-expression to brand growth.
   - Most creators experiment with both long and short videos, and noted that reposting long content as short videos does not necessarily drive views.
   - Some creators focus exclusively on short videos because long-form content requires more scripting and editing adjustments, making it harder to produce consistently.
2. Challenges in Video Creation
   - Long videos require extensive scripting, planning, and editing to maintain engagement, with some creators structuring content using chapters or subtitles.
   - Short videos are faster to produce but have unpredictable performance due to rapidly shifting trends.
3. Strategies for Maximizing Engagement
   - Creators use questions, anecdotes, or trending elements to capture viewers' interest within the first few seconds or in the middle of a video.
   - Memes, music transitions, and visual effects help maintain a dynamic pace.
   - Some creators use analytics (watch time, drop-off points) to refine content, while others rely on direct audience feedback (comments).
4. Perceptions of Audience Attention Span
   - Most creators do not believe that decreasing attention spans alone cause lower engagement. Instead, they attribute drops in retention to content quality, pacing, and relevance.
   - Mid-length videos (3–6 minutes) struggle as they lack the depth of long videos and the quick appeal of short ones.
   - Some creators noted a shift in audience preference toward shorter content, but they also observed that engaging long-form videos still retain dedicated viewers.

3.3 Process Map

Our process map illustrates the content creation journey, highlighting the often-overlooked storytelling and engagement strategies that drive audience retention.

1. The Visible Layer: Surface-Level Metrics. At the top of the iceberg is what we easily see—views, comments, and audience reactions. These metrics reflect the final product but don't reveal why content succeeds or fails.
2. The Mid-Layer: Production Efforts. Below the surface are scriptwriting, editing, and publishing—the tangible steps creators manage. While crucial, they don't guarantee engagement on their own.
3. The Deepest Layer: Storytelling and Engagement Strategy. At the core of content success lies storytelling and engagement strategy, which shape audience retention but are harder to measure. Storytelling makes content compelling and emotionally engaging. Engagement strategies (pacing, structure, interactivity) influence how long viewers stay.

Creators and audiences often focus on surface-level results, but understanding and refining storytelling and engagement strategies is what truly drives success. Our system bridges this gap by offering insights that help creators optimize their content beyond basic metrics.

3.4 Story Board

Our storyboard illustrates the content creation journey and how our system helps creators visualize potential issues early and refine their scripts before filming.

1. Initial Script Writing: The content creator drafts a script but feels uncertain—will it be engaging enough? Will anything be missing? They recall past struggles where they only realized missing or weak sections after reaching the editing stage.
2. Pre-Filming Uncertainty: Before shooting, they hesitate—should they trust the script as is? They worry about realizing too late that some segments are missing or that certain parts won't hold audience attention.
3. Using the System and Refining the Script: With visualization and actionable suggestions, the creator previews engagement gaps and script flow before filming. This helps them identify weak areas early, adjust their content accordingly, and gain confidence before production.

This process highlights how our system bridges the gap between scripting, filming, and engagement strategy, allowing content creators to refine their work efficiently before moving to production.

4 Design goals

Based on our initial user research and clearly defined problem statement, we established three primary design goals for our system, each directly targeting specific challenges faced by content creators:

Provide actionable suggestions to improve engagement: Interviews revealed content creators frequently struggle to maintain viewer engagement, especially in long-form videos. Currently, they rely heavily on intuition or trial-and-error without clear guidelines. For instance, one creator expressed uncertainty regarding the optimal placement and frequency of memes or special effects. Another highlighted difficulties keeping viewers interested during long, factual segments, which often result in audience drop-off.

To directly address this issue, our system includes a **"Generate Suggestions" feature**. This analyzes scripts and provides targeted, actionable recommendations, such as:

- Specific paragraphs where storytelling, humor, or visuals should be inserted.
- Explicit advice on restructuring dense information sections to maintain viewer interest.

By clearly guiding creators, the system aims to reduce guesswork, enabling efficient content optimization.

Visualize Script Composition: Creators often discover structural issues like overly dense informational segments too late—during editing, when making significant adjustments is challenging. For example, a content creator noted they typically only realize that sections are excessively factual once in the editing stage, limiting their flexibility for adding engaging elements.

To help creators identify such structural issues early, our design includes a **color-coded timeline visualization**. This feature automatically categorizes each paragraph type (e.g., informational, storytelling, comedic, visual presentation) clearly and intuitively. Benefits include:

- Quickly spotting sections lacking engagement elements.
- Making informed decisions on content adjustments before committing significant editing resources.

Ensure an Easy-to-Use and Intuitive Workflow: Given that many interviewed creators are not full-time professionals, simplicity and seamless integration into existing workflows were crucial design considerations. Creators emphasized needing tools that offer quick, actionable insights without additional complexity. They highlighted the iterative nature of scripting, noting that one-time feedback is insufficient, as scripts typically undergo multiple revisions.

To meet these requirements, we designed an interface inspired by Grammarly—a familiar, intuitive layout:

- **Left side:** Simple text input for the script.
- **Right side:** A user-defined goal prompt clearly indicating content objectives.
- A prominent **"Check Script" button** generates understandable suggestions and a clear script visualization.
- Support for iterative workflow—allowing users to repeatedly edit their scripts and easily regenerate updated suggestions.
- Hover explanations and placeholders to guide users in interpreting feedback, making the tool accessible even to non-professionals.

This design ensures creators receive ongoing, actionable feedback without needing significant additional effort or learning curves.

5 System Design

We designed our system, ScriptPulse, which follows all of our design goals. The program follows a standard web-app structure, containing a frontend that the user interact with and a backend that process user input and requests.

5.1 System Workflow

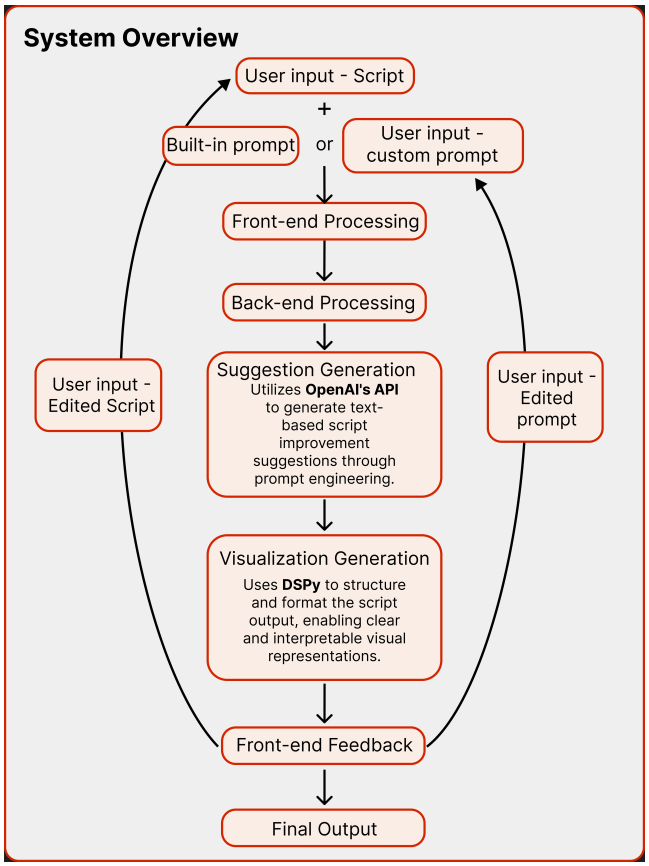

Figure 6. Standard Workflow of ScriptPulse

The overall system workflow is visually represented in Figure 6. The process begins with users inputting their video script, accompanied by either a default built-in prompt or an optional custom prompt provided by the user.

Upon submission, the script undergoes initial processing by the frontend, preparing and formatting the data for the backend. Next, the backend initiates two key processes:

- **Suggestion Generation:** Leveraging OpenAI's API, the backend generates targeted textual suggestions to enhance the script's quality. This step is driven by structured prompt engineering, carefully customized to user objectives defined either by the default or custom prompt.
- **Visualization Generation:** Utilizing DSPy, the backend system structures and categorizes script segments through LLM. Each paragraph is classified (e.g., informational, comedic, storytelling), enabling clear and visually interpretable representations of script composition and pacing.

Once generated, both suggestions and visualizations are delivered back to the front-end interface, where users can review detailed feedback and easily identify areas for improvement. The system supports iterative refinement: users can edit their scripts or prompts based on provided suggestions, subsequently repeating the analysis process. This iterative feedback loop continues until users finalize their script, achieving a polished output aligned with their content goals. This structured workflow streamlines script refinement, allowing users to easily visualize content distribution, incorporate actionable suggestions, and iteratively enhance their scripts based on insightful analysis.

5.2 Frontend Design

5.2.1 Introduction. Below is an in-depth write-up describing our **frontend design** and how it integrates with the underlying FastAPI + DSPy + OpenAI backend. This write-up walks through the rationale behind our React implementation, the main user flows, the code structure, and key UI decisions, offering a complete picture of how our script-analysis tool comes together on the client side.

5.2.2 Overview of the Frontend. Our frontend is built with **React** and styled with **Material UI (MUI)** to create a cohesive user experience. The application is designed around a central idea: users provide a script (either via direct text input or file upload), optionally supply a custom prompt (guidance or instructions), and then submit this data to our backend for analysis. Once the analysis is complete, the user receives categorized feedback—visualized clearly—and has the option to revise or store versions of their script for later reference.

Key Design Requirements.

1. **Ease of Use:** We wanted a frictionless interface that allows users to quickly input or upload scripts. By offering two modes (manual input and file upload), our interface caters to different user preferences.
2. **Rich Visual Feedback:** Categorized results should be immediately visible, so we use color-coded highlighting in the script text. This helps users see which segments are informational, comedic, storytelling, and so on.
3. **Version Control:** Users often iterate on scripts multiple times. Our design includes a built-in mechanism to save and load script versions. This feature keeps a complete edit history and helps maintain creative flow.
4. **Seamless Communication With the Backend:** The frontend consumes an HTTP POST endpoint (at `/check_scr` offered by **FastAPI**. All requests and responses adhere to a well-defined structure using Pydantic models, ensuring reliability and ease of parsing on both sides.

5.2.3 Layout and Flow. At a high level, the interface is laid out using **MUI's Grid** system to create two primary columns:

- **Left Column (Script Editing & Display)**
- **Right Column (Prompt Entry, Analysis Results, and Version History)**

Editing State: Users can choose "Manual" mode, which provides a standard multiline `TextField` component for typing or pasting script content directly (Figure 7). They also have an option to upload images into the text as placeholders. In "File Upload" mode (Figure 8), the application displays a drag-and-drop zone (or browse button) where users can upload text files of various formats (`.txt`, `.md`, `.docx`, `.json`). A `FileReader` reads the text and sets it in our local state (`script`). Once uploaded, the contents of the file appear as a preview.

- If the user chooses manual entry, they can freely type or paste text.
- If the user chooses file upload, the system reads the file in and populates the script area.

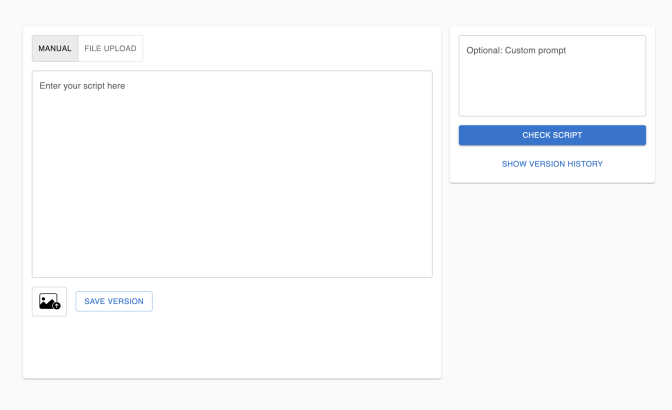

Figure 7. Manual input mode for entering video scripts.

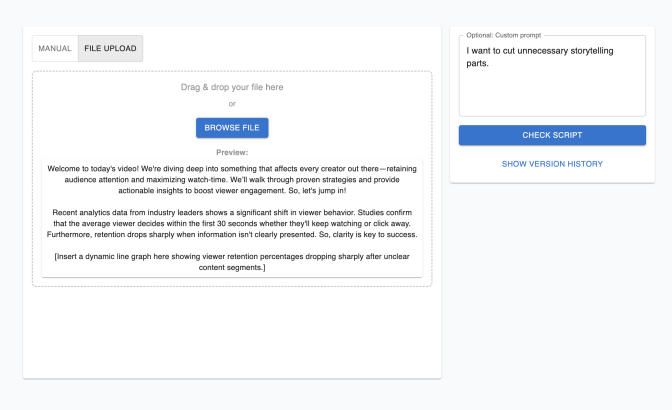

Figure 8. File upload mode and optional custom goal prompt.

Read-Only State: After the script is submitted for analysis and results come back from the backend, the script text is replaced by a highlighted read-only view. This highlighted script visually encodes each paragraph's category (e.g., comedic, informational, storytelling) with a color or gradient. In this mode, the user can only view the final colored text, ensuring they cannot accidentally modify the script while analyzing it.

Image Insertion: We allow an image file upload through a small icon button. When pressed, the chosen image is represented with a placeholder string ("`[image inserted]`"). This helps the user keep track of where visuals might be inserted in a final script without storing or rendering the actual images in the text area.

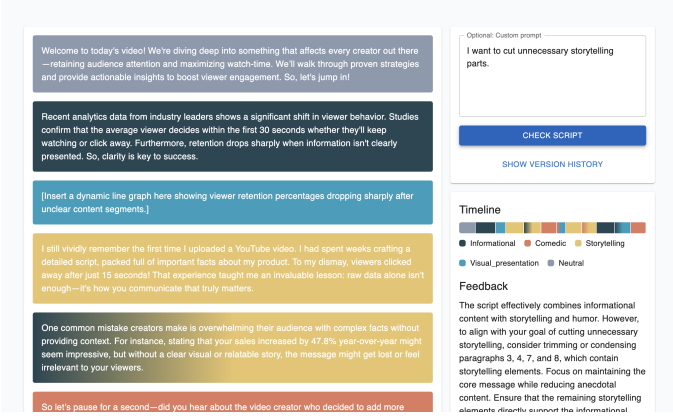

Figure 9. Analysis results showing categorized, highlighted script paragraphs and feedback.

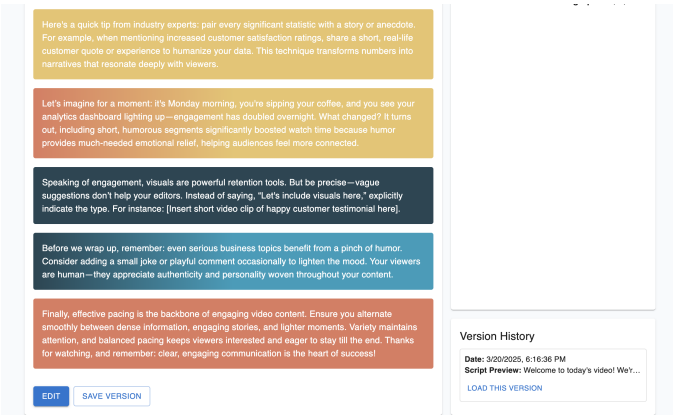

Figure 10. Bottom portion of analysis results with version history.

Version Control: While editing a script (in manual mode) or reviewing it post-upload, the user can click the **"Save Version"** button. This triggers the creation of a version snapshot as shown in the bottom of Figure 10 that's stored in the frontend state. Each version includes:

- A unique ID (based on a timestamp).
- The date/time when it was saved.
- The script text itself.
- The user's optional custom prompt at the moment of saving.

These versions appear in a **Version History** panel on the right column. Any previously saved version can be reloaded into the editor, effectively letting the user revert to an earlier draft.

Custom Prompt: We provide an optional `TextField` labeled "Custom prompt" that appears at the top of the right column. This text field captures user-specific instructions or goals that guide the analysis (e.g., "I want to cut unnecessary storytelling parts."). If provided, the backend merges this custom prompt with the script content to produce more targeted analysis.

Analysis: Below the Custom Prompt field, a **"Check Script"** button triggers the main action:

1. The script text and the prompt are posted to our FastAPI endpoint.
2. We anticipate a structured JSON response containing the LLM's classification and feedback.
3. If the script is empty or the request fails, we display an error message.

Analysis Results & Timeline: Upon successful analysis, the backend returns an object (referred to as `suggestions` in the code) containing:

- `feedback`: General feedback about the script and advice for improvements.
- Lists of paragraph indices for categories such as `information`, `comedic`, `storytelling`, `visual_presentation`, etc.
- A `duration` field estimating the time each paragraph might take in a final video.

As presented in Figure 9, we use a `Timeline` component to visually present the duration data for each segment. This timeline helps users see how long each part of the script might be when spoken or recorded, providing real-world insight into pacing.

Below the timeline, we display a short summary of the feedback text plus the recognized paragraph indices in each category. This breakdown is crucial for quick reference: the user can correlate these indices with the color-coded paragraphs on the left side.

Version History Panel: When the user wants to see previous script states, they click **"Show Version History"**. We render the saved versions in a list, each containing:

- The date/time stamp of when it was saved.
- A short preview of the script content (truncated).
- A button to **"Load This Version"** which immediately reverts the main editor to that version's script and prompt.

This approach ensures that iterative script editing is easy and transparent, letting users track changes over time.

5.2.4 Code Structure and Components.

1. **App.js:** Manages the top-level state: `script`, `userPrompt`, `suggestions`, `error`, etc. Determines whether the interface is in editing mode or read-only mode (based on `isReadOnly`). Houses the mode toggle (Manual or File Upload) and orchestrates how the UI changes with user actions. Implements version control logic (saving, listing, and loading versions). Sends AJAX requests (via **Axios**) to the backend endpoint to initiate analysis and handle responses.
2. **HighlightedScript.js:** Renders the script in a read-only, color-coded style. Splits the script into paragraphs and checks each paragraph's index against the lists in `suggestions`. Applies different background colors (or a gradient, if categories overlap) to visually signify comedic, informational, storytelling, etc.
3. **Timeline.js:** Presents a timeline visualization for analyzed paragraphs. Typically shows how many seconds each paragraph might take and can use bars or other designs to illustrate distribution.
4. **Material UI Elements:** We rely on MUI's `Grid`, `Container`, `Paper`, `Box`, `TextField`, `Typography`, and `Button` to create a responsive layout with a consistent style. The color scheme and spacing align with MUI defaults, simplifying design and ensuring a polished look.

5.2.5 Why This Approach?

- **Speed and Reliability:** Using React hooks and local state ensures a fast and interactive experience. FastAPI is equally performant on the server side, offering low-latency responses.
- **Scalability:** If we expand categories or add more complex timelines, we can do so without breaking existing logic. The combined flexibility of React and the structured approach in the backend fosters rapid iteration.
- **Excellent Developer Experience:** Material UI greatly reduces styling overhead. Dedicated components like `HighlightedScript` and `Timeline` keep the overall architecture clean, while debugging is straightforward with well-typed data flows.

5.3 Backend Design

The backend server leverages **FastAPI** for robust API routing and communication with the frontend, while **DSPy** coupled with OpenAI's API handles script analysis and categorization.

Initialization: We start by creating a FastAPI application instance, which sets up a CORS middleware to allow the frontend (running on `localhost:3000`) to communicate with this backend. FastAPI was chosen due to its performance, ease of use, automatic documentation, and excellent integration with Pydantic

Model Definition: We defined `ScriptRequest(BaseModel)`, the class format of the shape of data the backend expects in a POST request:

- $script : str$ (required) – the textual content of the user's video script.
- $user\_prompt : Optional[str]$ (optional) – allows users to provide custom analysis instructions or goals, which guide the LLM towards personalized analysis outcomes.

This model is used for request validation, ensuring your endpoint receives properly structured JSON.

Endpoint: A single POST endpoint handles script analysis requests (`@app.post("/check_script")`). It reads the incoming request body into an instance of ScriptRequest, which automatically validates the input thanks to Pydantic. Inside the endpoint, the script text ($request.script$) and an optional user prompt ($user\_prompt$) are passed to $analyze\_script$ (API imported from $dspy\_utils$).

DSPy and OpenAI pipeline: We used DSPy [**?**] for easy prompting with structured input and output field. The analysis pipeline comprises three clear stages:

1. **Script Parsing:** A simple parser ($script\_parser$) splits the raw input script into discrete paragraphs by detecting empty newlines, preparing a list of paragraph strings to be analyzed individually.
2. **Prompt Construction:** Using DSPy's structured input fields, the parsed paragraph list ($script : list[str]$) and any custom user instruction ($user\_prompt$) are providedto the LLM. The custom user prompt is prioritized by the LLM during analysis, ensuring that the returned feedback aligns closely with the user's stated objectives.
3. **Output Categorization:** The LLM classifies each paragraph into predefined categories, strictly following the structured DSPy output fields:

We defined the output fields as such:

- **feedback** str: General suggestions and detailed feedback on the script.
- **information** $list[int]$: List of indices of paragraphs that are informationally dense.
- **comedic** $list[int]$: List of indices of paragraphs with comedic elements.
- **storytelling** $list[int]$: List of indices of paragraphs employing storytelling techniques.
- **visual_presentation** $list[int]$: List of indices of paragraphs that suggest visual presentation elements.
- **duration** $list[float]$: Estimated duration (in seconds) for each paragraph in the final video.

Indices of paragraphs that the LLM classifies as one specific category would be outputed as a new list. (e.g. output list

information would contain all indices of paragraphs that the LLM belives are informationally dense). This structured approach provided by DSPy ensures accurate and predictable output formats, crucial for seamless frontend integration and visualization.

6 Evaluation

Evaluation question, methods, and analysis approach, incorporating instructor feedback

6.1 Motivating Questions

How useful and actionable do users find the suggestions and visualizations?

6.2 Methods

We conducted a mixed-methods study with 15 participants. Each participant was asked to use the system to revise a script, either one they had previously written or a provided sample script. The process was divided into four key stages:

- Exploration Phase – Participants familiarized themselves with the interface.
- Visualization Phase – Participants analyzed their script's structure using the timeline and highlights.
- Suggestion Phase – Participants reviewed AI-generated feedback to refine their content.
- Editing Phase – Participants modified their script based on the provided insights, and went through each phase again.

6.3 Analysis Approach

To answer our evaluation questions, we collected both quantitative and qualitative data:
Quantitative Metrics:

- Time spent in each phase – Indicates which features were most engaging.
- Number of interactions (visualization-script, suggestion-script) – Shows how frequently users engaged with feedback.

Qualitative Metrics:

- User comments and feedback – Captures perceptions of usefulness and engagement.
- Observations on interaction patterns – Helps understand user decision-making processes.

7 Findings

We conducted user evaluations with 15 participants, including two rounds of testing, to assess the usability, usefulness, and overall effectiveness of our system. Participants consisted of both experienced content creators and non-creators, providing diverse perspectives on system performance.

7.1 Qualitative Results

The qualitative data provided rich insights into how users perceived and interacted with the system. We conducted thematic analysis to categorize the qualitative feedback into four main themes:

Usability and User Interaction: Participants consistently mentioned the actionable and specific nature of the suggestions and visualizations as valuable. They appreciated the system's structured guidance, even when they chose not to follow every recommendation exactly. Some example feedbacks include:

- Participant 1: *"The suggestions are highly actionable, providing clear and detailed guidance. Even though I don't always follow them exactly, they still inspire new ideas."*
- Participant 2: *"The visualization helps prevent issues ahead of time, making my workflow more efficient."*

Some participants suggest improvement and what's lacking from the current system:

- Participant 6: *"Feedback could be improved by including concrete examples—if the suggestion says 'Add more anecdotes,' it should provide specific examples."*

Customization and Personalization: Some users found the system less helpful in aligning with their unique creative styles or specific content preferences. This highlighted the need for personalization or adaptive capabilities to better serve experienced creators. Some example feedbacks include:

- Participant 3: *"It's useful, but not something I'd rely on heavily. The system doesn't yet learn or adapt to my unique content style."*
- Participant 8: *"I want more customization in user prompts so the suggestions can match my needs better.""*

Usability and User Interaction: Most participants felt the system was intuitive and quickly grasped how to use it. However, several suggested improvements to further streamline usability, including clearer labeling and better navigation features. Some example feedbacks include:

- Participant 2: *"The sidebar for suggestions should be wider to better display feedback alongside the script."*
- Participant 4: *"Hover effects on paragraphs would improve usability—hovering over suggestions or timeline segments should highlight corresponding paragraphs."*
- Participant 7: *"The fact that paragraph indices start at 0 is unintuitive—this should be fixed for general users.""*

Broader Applicability and Long-Term Use Potential: Participants identified potential uses for the system beyond video creation. Particularly, non-creators found value in using the tool for scriptwriting in general, such as speeches or presentations. Some example feedbacks include:

- Participant 4: *""Even as a non-creator, I found the system understandable and useful. It could easily apply to public speaking or presentations.""*

- Participant 9: *"Integrating multimedia like images and audio could expand its usefulness beyond just scripts."*

7.2 Quantitative Results

Quantitative data, gathered through measuring user interactions and time spent across various stages, provided measurable evidence supporting the qualitative observations.

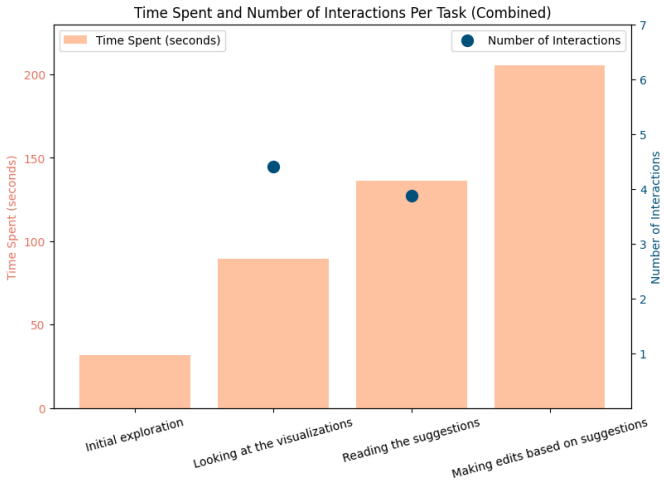

Figure 11. Bottom portion of analysis results with version history.

Time Spent per Task: Participants spent varying amounts of time in each stage, reflecting differences in familiarity and interaction patterns as shown in Figure 11:

- **Initial Exploration:** On average, users quickly understood system functionality (30 seconds).
- **Visualization Analysis:** Users spent an average of 106 seconds analyzing visualizations.
- **Suggestions Reading:** Users spent an average of 148 seconds carefully reviewing suggestions.
- **Script Editing:** The longest duration was dedicated to script editing, averaging 213 seconds, indicating active engagement with suggestions and visualizations.

Interaction Patterns: The frequency of interactions provided insights into the specific features that users found most useful:

- Visualization-script interactions averaged around 5 times per user, showing frequent toggling between script editing and visualization to ensure balanced content structure.
- Suggestion-script interactions averaged around 4 times per user, indicating deliberate and careful review of provided suggestions.

8 Discussion

Discussion of your takeaways based on the evaluation, limitations and future work, mistakes and lessons learned

8.1 Key Takeaways

Our user evaluations provided several important insights:

- **Effective Visualization and Suggestions:** Participants consistently found the visualization and suggestion features helpful, enabling targeted script improvements. The color-coded timeline clearly highlighted content structure, guiding users to balance information density and engagement. Suggestions were mostly actionable and inspired creativity, even if not always followed exactly.
- **Iterative Workflow:** Users greatly benefited from the system's iterative nature. Version control and file upload significantly improved workflow efficiency, enabling users to confidently experiment with script modifications, thus enhancing creative exploration and reducing editing anxiety.
- **Broad Usability and Accessibility:** The tool's intuitive interface allowed even non-content creators to quickly understand and leverage the system, suggesting broader applicability beyond video creation—for example, speeches or presentations.

8.2 Limitations and Future Work

We have also identified several limitations of our current system, many areas require further improvement in the future to better accomplish our design goal:

- **Customization and Adaptability:** Experienced creators expressed a need for more personalized suggestions that adapt to their specific styles. Future work could include user profiles with self-uploaded "portfolios of video scripts" or style-based adaptive feedback mechanisms to better tailor suggestions to individual preferences.
- **Interactive Enhancements:** A recurring usability concern was the non-interactive timeline and unclear paragraph referencing. Improvement's have been made on these, but future iterations should focus on more intuitive UI design and better user interactions to ehance readability and navigability.
- **Enhanced Multimedia Integration:** While placeholders for visual insertions were effective, direct integration of multimedia (e.g., images, audio, video clips) within the script editor was frequently requested. This would further streamline the workflow and make the tool more robust.
- **Feedback Clarity and Examples:** Some participants desired more explicit, example-driven feedback rather than general suggestions. Future work involves refining the AI model prompts to generate more concrete examples to improve usability and effectiveness.
- **Possible Integration with Editing Software:** To better integrate this tool into content creator's standard workflow and adapt different persons' preferences, we

can also investigate about the possibility of integrating our system into exist video editing softwares like Premiere or Final Cut Pro. Combining with multimedia input, this addition will maximize the context-awareness of the AI agent, allowing them to provide better suggestions. It also helps to expand the usage context to also include post-shooting and editing phase, not limited to script writing phase.

8.3 Mistakes and Lesson Learned

- **AI Output Clarity and usefulness:** The initial design prompted the AI model to provide generalized feedback without sufficient context-specific examples. Evaluations revealed users strongly preferred actionable and specific feedback with concrete examples. Future iterations will prioritize clearer and more structured AI output to meet user expectations.
- **Workflow Flexibility:** Initially, we did not anticipate the high value users placed on version control. After implementation, this feature significantly improved user confidence during editing.
- **Importance of Incremental Improvements:** Early attempts to incorporate many features simultaneously resulted in UI complexity and confusion. We learned to prioritize incremental, user-driven changes, ensuring each feature meaningfully enhanced user experience rather than adding complexity.

8.4 Conclusion

Overall, our evaluations indicate that the system is highly beneficial in supporting structured, iterative script improvements. However, addressing customization, multimedia integration, and clearer, example-driven feedback are essential next steps. The insights gained emphasize the value of iterative development guided by user feedback, encourage future iterations to be more effective.

References

[1] Chiu, X., et al. 2021. "Evaluating Screenplay Quality with Linguistic and Narratology-Inspired Features." In *Proceedings of the 15th AAAI Conference on Web and Social Media (ICWSM)*, 112–120.

[2] Simmons, J. 2020. "Detecting Narrative Turning Points in Screenplays using Machine Learning." *Computational Linguistics and Storytelling*, 9(4): 35–48.

[3] Fong, E., and Gui, F. 2022. "LLM-Driven Narrative Engagement: Predicting Readers' Surprise and Continuation." *Transactions of the Association for Computational Stories*, 5(2): 56–67.

[4] Zhao, T., Yin, T., and Yang, R. 2021. "Analyzing Online Video Engagement: Production, Pacing, and Retention." In *Proceedings of the 25th International Conference on Multimedia*, 2217–2225.

[5] Yang, Y., et al. 2022. "Wordcraft: GPT-3 for Collaborative Story Writing." *CHI '22 Extended Abstracts on Human Factors in Computing Systems*, 1–6.

[6] Mathewson, K., et al. 2022. "Dramatron: Hierarchical Language Model Prompting for Coherent Script Generation." *arXiv preprint*, arXiv:2210.06742.

[7] Jasper. 2022. "Jasper AI Writing Tool." https://www.jasper.ai/, accessed October 2022.