iScore: Visual Analytics for Interpreting How Language Models Automatically Score Summaries

Adam Coscia
Georgia Institute of Technology
Atlanta, Georgia, USA
acoscia6@gatech.edu

Langdon Holmes
Wesley Morris
langdon.holmes@vanderbilt.edu
wesley.g.morris@vanderbilt.edu
Vanderbilt University
Nashville, Tennessee, USA

Joon Suh Choi
Georgia State University
Atlanta, Georgia, USA
jchoi92@gsu.edu

Scott Crossley
Vanderbilt University
Nashville, Tennessee, USA
scott.crossley@vanderbilt.edu

Alex Endert
Georgia Institute of Technology
Atlanta, Georgia, USA
endert@gatech.edu

ABSTRACT

The recent explosion in popularity of large language models (LLMs) has inspired learning engineers to incorporate them into adaptive educational tools that automatically score summary writing. Understanding and evaluating LLMs is vital before deploying them in critical learning environments, yet their unprecedented size and expanding number of parameters inhibits transparency and impedes trust when they underperform. Through a collaborative user-centered design process with several learning engineers building and deploying summary scoring LLMs, we characterized fundamental design challenges and goals around interpreting their models, including aggregating large text inputs, tracking score provenance, and scaling LLM interpretability methods. To address their concerns, we developed *iScore*, an interactive visual analytics tool for learning engineers to upload, score, and compare multiple summaries simultaneously. Tightly integrated views allow users to iteratively revise the language in summaries, track changes in the resulting LLM scores, and visualize model weights at multiple levels of abstraction. To validate our approach, we deployed *iScore* with three learning engineers over the course of a month. We present a case study where interacting with *iScore* led a learning engineer to improve their LLM's score accuracy by three percentage points. Finally, we conducted qualitative interviews with the learning engineers that revealed how *iScore* enabled them to understand, evaluate, and build trust in their LLMs during deployment.

CCS CONCEPTS

• **Human-centered computing** → **Visual analytics**; • **Computing methodologies** → **Neural networks**; • **Applied computing** → *Interactive learning environments.*

This work is licensed under a Creative Commons Attribution International 4.0 License.

IUI '24, March 18–21, 2024, Greenville, SC, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0508-3/24/03
https://doi.org/10.1145/3640543.3645142

KEYWORDS

Data visualization, visual analytics, large language models, explainable AI, educational technology

ACM Reference Format:
Adam Coscia, Langdon Holmes, Wesley Morris, Joon Suh Choi, Scott Crossley, and Alex Endert. 2024. iScore: Visual Analytics for Interpreting How Language Models Automatically Score Summaries. In *29th International Conference on Intelligent User Interfaces (IUI '24), March 18–21, 2024, Greenville, SC, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3640543.3645142

1 INTRODUCTION

The advent of large language models (LLMs) has catalyzed state-of-the-art research in the learning analytics community on advancing the capabilities of adaptive educational tools, namely automated scoring of summary writing [6, 34]. For example, LLMs can be used to automatically score a summary written on a larger body of text (Fig. 1) in a variety of learning environments. For data scientists in the learning analytics community, henceforth called learning engineers, it is extremely important to test LLMs on many different summaries and understand how the LLMs work. However, using deep learning models introduces opaqueness into model evaluation [28, 40], making it difficult for learning engineers to close the loop of model development [58]. Interactively exploring how their LLMs score different summaries can help learning engineers understand the decisions on which the LLMs base their scores, discover unintended biases, update the LLMs to address the biases and mitigate the potential pedagogical ramifications of prematurely deploying untested LLM-powered educational technologies [27].

Understanding and evaluating LLMs is extremely challenging due to their unwieldy size and ever-growing number of parameters, making it difficult to identify the causes of performance issues and address them as needed [46]. Transparency is critical for building trust in using LLMs [30], especially for learning engineers who are increasingly using LLMs as "black boxes" for downstream tasks, as well as during human-in-the-loop evaluation of LLM performance where quantitative benchmarks often fall short [3]. Visual analytics tools are increasingly used for improving the transparency of LLMs and helping developers interpret LLM behavior [24, 31]. Yet little research at the nexus of machine learning (ML) and educational

data has leveraged visual analytics for explaining ML-powered educational technologies [10, 19, 35, 65] and none have used LLMs, presenting an opportunity to collaboratively develop design principles in this space with domain experts. We seek to build a visual analytics system that makes evaluating summary scoring LLMs more transparent for learning engineers, helping them understand and build trust in their LLMs before deployment.

Through a user-centered design process [43] working directly with learning engineers who train, test, and deploy automatic summary scoring LLMs (Sect. 3), we synthesized several fundamental challenges around the aggregation, provenance, and scalability of visualizing writing and LLM scoring data together. For learning engineers, characterizing quality in writing samples involves comparing differences in both text and LLM scores across multiple scoring dimensions and between multiple different samples, revisions of the same sample, and expert-scored "ground truth" samples. Performing these tasks enables them to calibrate their trust in the LLM summary scores. Increasing transparency in LLM scores requires probing model behavior externally by varying input parameters and internally by exploring model weights, as well as scaling these methods to large amounts of text while keeping learning engineers in the loop at different levels of detail. To address these challenges in a single interface, a successful visualization system should help learning engineers scaffold the evaluation process around comparing revisions of multiple writing samples, scale multiple interpretability methods to work with large inputs and visually aggregate text at multiple levels of abstraction.

We present *iScore* (Sect. 4), an open-source[1], interactive visual analytics tool for learning engineers to upload, score, and compare multiple summaries of a source text simultaneously. *iScore* introduces a new workflow for comparing the language features that contribute to different LLM scores by structuring analysis across three coordinated views. First, users upload, score and can manually revise and re-score multiple source/summary pairs simultaneously in the *Assignments Panel*. Then, users can visually track how scores change across revisions in the context of expert-scored LLM training data in the *Scores Dashboard*. Finally, users can compare model weights between words across model layers, as well as differences in scores between automatically revised summary perturbations, using two model interpretability methods in the *Model Analysis View*. Together, these views provide learning engineers with access to multiple summary comparison visualizations and several well-known LLM interpretability methods including attention attribution, input perturbation, and adversarial examples. Combining these visualizations and methods in a single visual interface broadly enables deeper analysis of LLM behavior that was previously time-consuming and difficult to perform.

To validate our approach, we deployed *iScore* with our collaborators, a learning analytics team, over the course of a month, and conducted follow-up interviews with the same engineers from the team with whom we collaboratively designed *iScore* (Sect. 6). We first describe a case study where one of the learning engineers used *iScore* to improve the accuracy of LLMs used in *iTELL*, an intelligent textbook framework that can automatically score summaries of textbook sections written by learners such as students. Using feedback from *iScore*, the engineer improved the scoring accuracy of their LLM by three percentage points. From the same experts' qualitative feedback, we report several findings. *iScore* improved understanding of how specific models work; e.g., removing the first sentence of a summary caused some models' scores to drop by up to 90%. The model analysis visualizations helped engineers "see" what the LLMs paid attention to, while tracking changes in scores across revisions illustrated how adversarial examples can trick LLMs into giving incorrect scores. The perturbation visualizations were unanimously considered the most useful feature for inspiring trust when deploying LLMs in critical learning environments. Reflecting on the case study and findings, we then discuss implications for the design of future systems, lessons learned on building responsible and ethical AI for education, generalizing our techniques to other LLMs and finally limitations and future work (Sect. 7).

In summary, our paper contributes: (1) design challenges and user tasks for helping learning engineers evaluate automated summary scoring LLMs; (2) *iScore*, an open-source visual analytics tool that aggregates and compares LLM data and LLM-scored writing samples; (3) a case study detailing how learning engineers deployed *iScore* to improve their LLMs and usage scenarios that demonstrate the generalizability of *iScore*; and (4) a qualitative evaluation with learning engineers describing how *iScore* helped them understand, evaluate, and build trust in their LLMs during deployment.

2 RELATED WORK

2.1 Automatically Scoring Summary Writing

Summary writing is a valuable pedagogical tool to help learners build knowledge about a subject area, as well as for assessment purposes [21, 23, 38]. A meta-analysis of 56 experiments found that text summarization improved learning regardless of the knowledge domain [22]. The reason for these learning increases may be that, when summarizing, learners are asked not only to retrieve information from the text but also to construct and organize their own schemata of the content area [17, 36, 45]. Despite the benefits, scaling summarization tasks in learning environments is challenging; providing feedback to learners is both difficult and time-consuming [18]. In response to this challenge, researchers have developed several strategies to automatically score summaries. For example, Crossley et al. used indices of linguistic features in summaries as well as Word2vec similarity scores to develop a model that explained 53% of score variance [12]. Deep learning methods are particularly well-suited to this task, as Botarleanu et al. demonstrated by using fine-tuned large language models (LLMs) to explain 55% of the score variance in their dataset [6]. Morris et al. improved on this work with LLMs by using principal components rather than raw scores to explain 66 – 79% of score variance [34]. While the language features used to explain summary scoring mapped onto expectations about what makes a good summary (i.e., overlap with source text, text organization, and vocabulary), the language features that power the LLM summary scoring models are very difficult to interpret.

2.2 Modeling Language With Transformers

We aim to help learning engineers interpret large transformer-based language models that automatically assign scores to written summaries of source text. Language models learn to model the

[1] *iScore* models and code: https://github.com/AdamCoscia/iScore

probability of a token occurring in a sequence (e.g., a word in a sentence). Transformer-based language models work by encoding all input words in a sentence into numeric representations known as embeddings. Attention mechanisms are then used to combine different word embeddings in a sentence together, creating new embeddings that are contextually informed by different parts of the sentence [52]. Attention weights determine how different embeddings are combined, and they have been used to explore how transformers work [11]. Large transformer-based language models such as BERT [14] and GPT-3 [8] have demonstrated state-of-the-art performance on a variety of tasks [46] in part due to pre-training via self-supervised learning using large-scale unlabeled document corpora. Pre-training captures foundational knowledge of semantic and syntactic relationships in baseline models useful for fine-tuning on specific tasks downstream with fewer examples needed.

The models used in our work build on Morris et al. [34] by adopting the Longformer [2] architecture. Longformers are an embedding transformer model based on RoBERTa [32] that tokenize a sequence of words as input, replace each token with a 768-dimensional embedding in a semantic vector-space, sum the embeddings with positional embeddings to encode relative position information, and finally feed the embedding matrix into a neural network model. The model comprises 12 attention layers in which the input embedding matrix is transposed and multiplied by itself to form similarity metrics pairwise between embeddings, or **attention weights**. For most BERT-style models, each token at each attention layer is attended to by each other token, resulting in higher demands for computation as the input sequence length increases. As a result, these models have a limited max sequence length typically around 512 tokens. However, providing additional context by including source text as input during LLM training increased performance compared with training on summaries alone [34]. To solve input sequence length limitations, Longformer assigns **global attention** to one or more tokens and a **sliding attention window** which moves across the text. Tokens in global attention are attended to by every other token while tokens in the sliding attention window are attended to only by other tokens within the window. As a result of this design, Longformer can accept much longer texts, up to 4096 tokens, while remaining computationally efficient. To increase model interpretability, *iScore* addresses novel challenges around visualizing the sliding window, global attention, and the increased number of tokens and attentions at scale.

2.3 Interpreting ML Using Visual Analytics

Visual analytics is an increasingly popular approach for analyzing and interpreting machine learning (ML) models [24, 31]. We build on a rich history of visual analytics tools for understanding educational data [16, 53, 64], drawing inspiration specifically from systems that seek to visually explain ML-powered educational technologies. In this space, Mubarak et al. built a system for visualizing patterns of learner interactions with videos in Massive Open Online Courses (MOOCs) to help explain predictions of learners' weekly performance from the interaction data using deep learning models [35]. Chen et al. engaged a user-centered design process with instructors and ML researchers to design and develop DropoutSeer [10], a visualization system for visually explaining ML predictions of learner dropout in MOOCs. Garcia-Zanabria et al. utilized a similar approach to visually explain predictions of learner dropout using counterfactual explanations in their system, SDA-Vis [19]. Zhang et al. leverage deep learning models (CNN and LSTM) to predict learner dropout and visualize counterfactual explanations using DropoutVis [65]. A common theme that cuts across these tools is a focus on human-in-the-loop workflows that enable data scientists like learning engineers to inject human intuition and domain expertise into the iterative process of training and validating model performance [58], which *iScore* makes heavy use of.

To increase transparency for learning engineers, we aim to visualize LLM performance across multiple interpretability methods. One method is directly visualizing a model's internal architecture as a form of explanation [48, 60]. With transformers, several tools focus on visualizing the internal prediction process of transformers as changes in attention weights across each layer and head of a model [13, 25, 54, 59]. However, there is debate as to whether attention weights in transformers can be used as a source of interpretation for model performance [1, 11, 26, 61]. Alternatively, structuring visual comparison between variations in model inputs and resulting outputs presents a more flexible and model-agnostic analysis approach. For example, VizSeq is a visual analysis toolkit for interactively evaluating language model task benchmarks [57]. Other tools present a visual analytics workflow to analyze changes in language model weights under various task-specific scenarios [20, 50]. By combining internal and external interpretability techniques in *iScore*, we seek to give users more control and flexibility over how they interact with the models, increasing understanding of model performance using alternative perspectives [30].

3 DESIGN PROCESS

Our goal in this work is to build a visual analytics system that enables data scientists in the learning analytics community, i.e. learning engineers, to interpret how large language models (LLMs) automatically score summary writing, helping them calibrate their trust in these models before deploying them. To do this, we engaged in user-centered design methodologies including contextual inquiry, rapid prototyping, and design iteration. Our team comprised all authors and included visualization experts in human centered computing and interaction design as well as learning engineers with expertise in natural language processing (NLP) and developing learning tools. Over the course of several months, we worked together in multiple virtual formative sessions to develop a shared understanding of the pain points in maintaining a human-in-the-loop workflow for monitoring and improving LLMs used for automatic summary scoring.

We first describe the workflow of our collaborators and their development of summary scoring LLMs (Sect. 3.1) presented as examples throughout the paper. We then summarize the key design challenges and user tasks (Sect. 3.2) for interpreting how these models are performing that our visual analytics solution addresses.

3.1 Background: Summary Scoring LLMs

In this paper, we demonstrate *iScore* using summary-scoring LLMs that our collaborators specifically developed for *iTELL*. In Sect. 6.1, we present a case study with a learning engineer from our team

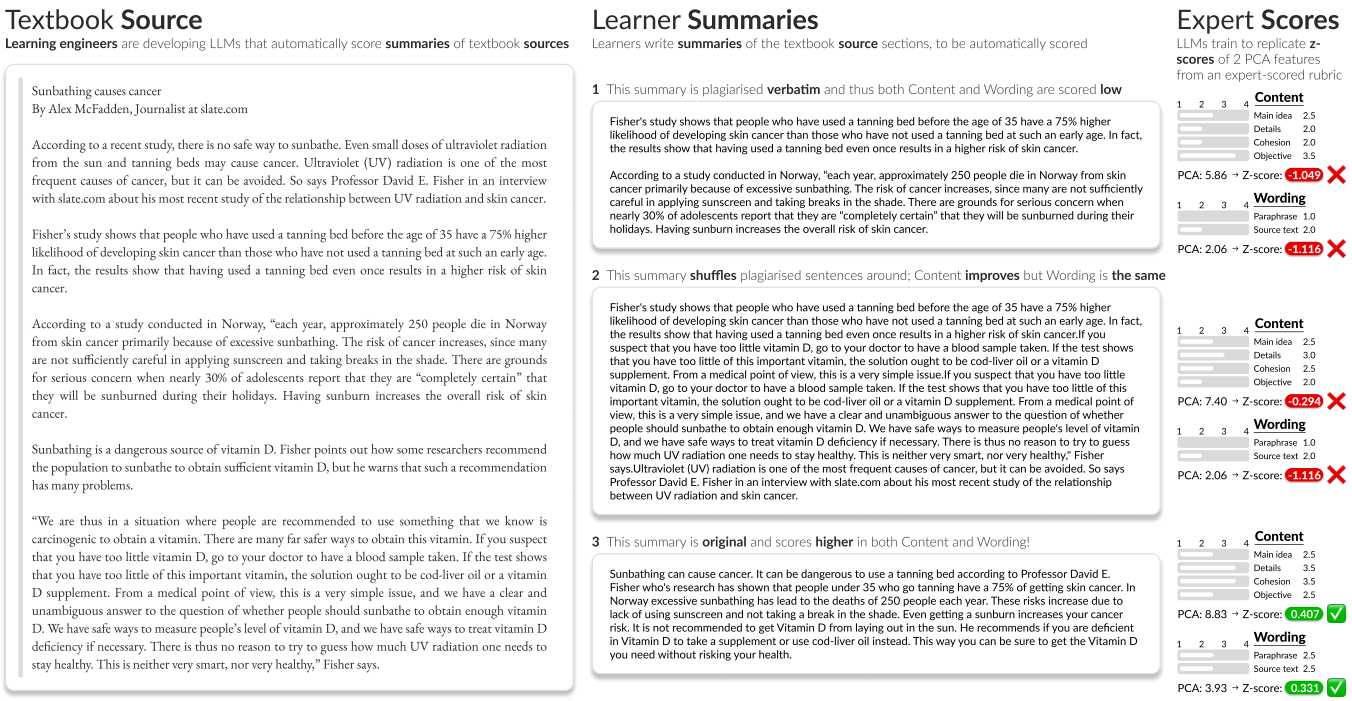

Figure 1: An example of a textbook source, learner summaries and expert scores used to train the LLMs visualized in *iScore*. Content and Wording LLMs each assign a continuous score that represents components of an analytic rubric. Learning engineers seek to characterize how changes in scores relate to differences in summaries via comparison. *iScore* provides inputs for multiple summaries per source and visualizes their predicted scores simultaneously in context of the "ground truth" training data.

who used *iScore* to improve the accuracy of LLMs in *iTELL* by three percentage points. By scaffolding evaluation around pairs of source and summary text, *iScore* presents a new workflow for learning engineers to interact with summary scoring LLMs.

3.1.1 iTELL: Textbooks That Score Summaries. Computational advances at the intersection of artificial intelligence (AI) and natural language processing (NLP) have catalyzed interest in developing intelligent textbooks that present adaptive "smart" functionalities to learners such as personalized feedback [9]. To help instructors quickly and simply build intelligent digital textbooks at scale, our collaborators developed *iTELL* (Intelligent Texts for Enhanced Lifelong Learning) as a framework for converting static text into a dynamic interactive web application with minimal labor and technical expertise required from content creators. A key intelligent feature built into *iTELL* is having learners summarize the section that they have read at the end of each textbook section. *iTELL* then provides formative feedback on the **Content** and **Wording** of these summaries using two LLMs, one for each facet.

3.1.2 Sources, Summaries and Scores. Our collaborators trained two BERT-style Longformer [2] models that assign **Content** and **Wording** scores to **summaries** of textbook **source** content. After the case study, they trained two more models, **Content (global)** and **Wording (global)**. We evaluate all four models in this paper.

Fig. 1 shows how scores assigned by the LLMs can vary across revisions of three different summaries of the same textbook source. The scores are defined during LLM training based on a $1-4$ scaled analytic rubric with six criteria: (1) main idea, or to what extent the summary captured the main idea of the source; (2) details; (3) cohesion, or how well the summary was rationally and logically organized; (4) objective language, or reflecting the point of view of the source; (5) paraphrasing, or avoiding plagiarism by paraphrasing the original material; and (6) language beyond the source, or how well all relevant details were included in the summary. Training samples were first expert-scored according to the rubric. Then, to reduce the dimensionality of the LLM output, the rubric criteria for each training sample were distilled using a principal component analysis (PCA) into two components. Content comprises main idea, details, cohesion and objective language, while Wording comprises paraphrasing and language beyond the source. The two PCA scores for Content and Wording were then z-score normalized. In this way, the scores which the LLMs are trained to assign now represent the standard deviation of the scored summary from the mean score along a single dimension, transforming the rubric into a more interpretable and useful result. These scores are then used as a target label in the training data for each respective model.

The models were fine-tuned on 4690 different training summaries written on 101 different source texts, enabling them to adapt to source/summary combinations on different topics with minimal to no adjustments needed. The training task was modeled as a sentence classification task [56], where each model tries to predict a continuous score (i.e. the Content or Wording score) as a label. In

Table 1: Design Challenges (C) and User Tasks (T)

C1	Characterizing how summaries and scores are related
T1	Stratify scoring across several facets of writing
T2	Compare multiple summaries and scores at once
T3	Use expert-scored writing samples as a reference
C2	Comparing differences between versions of summaries
T4	Highlight how revisions affect assignment scores
T5	Investigate multiple different types of revisions
T6	Track the provenance of both the writing and scores
T7	Select and view previous versions of summaries
C3	Understanding model parameters and behaviors
T8	Compare variations in model inputs to model outputs
T9	Examine the internal model weights (i.e. attention)
C4	Bridging global and local model behavior
T10	Summarize interactions between all tokens at once
T11	Compare token, layer and head interactions
T12	Drill down to interactions at specific layers and heads

a typical multi-class classification task, the output is composed of multiple labels, each of which receive a logit score. During training, the model outputs are compared against one-hot encoded target outputs and loss is computed using cross-entropy. During inference, the max logit score is chosen as the predicted label. When predicting a continuous score, however, only a single label is used and the logit score of that label is interpreted as the predicted score. Instead of cross-entropy, mean squared error (MSE) was chosen as the loss function because MSE penalizes larger errors more than smaller errors, thus encouraging closer alignment with the predicted labels. This method is commonly used in tasks that require the prediction of a continuous label [29, 37, 47]. The models were trained for six epochs with a batch size of eight and a learning rate of $3e$-5.

3.2 Design Challenges and User Tasks

From our formative sessions, we synthesized several key design challenges **(C)** and user tasks **(T)** summarized in Table 1 and integrated into our descriptions of the system (Sect. 4), usage scenarios (Sect. 5) and evaluation (Sect. 6) throughout the rest of the paper.

C1 Characterizing the relationship between summaries and scores. Summaries balance multiple facets of writing quality (text length, cohesion, paraphrasing, details, etc.), making it difficult to automatically assign a single graded score encapsulating all aspects of importance. To address this, learning engineers first seek to model each of these facets individually by *stratifying scoring along several dimensions, using multiple scoring models* **(T1)**. This can help language models automatically assign more objective scores to specific facets such as plagiarism or grammar. With multiple scores, engineers can then begin to interpret and improve model performance by *comparing multiple summaries and scores simultaneously* **(T2)**. Fig. 1 provides an example of how differences in summaries (e.g., syntactic and semantic) can lead to differently assigned scores. To further enrich comparison, engineers also *use expert-scored writing samples as a reference* **(T3)**. In the context of our Content and Wording summary scoring models, these expert-scored samples are used as training data. Our interface should enable users to compare multiple summaries and scores at once, as well as in reference to the "ground truth" expert-scored training data.

C2 Comparing differences between versions of summaries. If a summary's scores are too low, revisions need to be made that address the factors that the models base their assessment on. It is critical to give learning engineers methods that *highlight how revisions affect automatic score assignments* **(T4)**. Engineers are also interested in *investigating multiple types of revisions that can include spelling and grammar, paraphrasing, and keywords* **(T5)**. Together, these methods can reveal insights into how the language models are working and provide engineers with useful criteria for feedback to learners as they write their summaries. As revisions accumulate, our system should *track the provenance of both the writing and scores* **(T6)**, and allow users to *select and view previous versions of summaries* **(T7)**. This compliments the need for comparison of multiple summaries and scores simultaneously in Fig. 1 by adapting the comparison to multiple versions of the same summary.

C3 Understanding model parameters and behaviors. Using a deep learning approach for automatically assigning scores introduces opaqueness into understanding model behavior [30], making it difficult to support learning engineers in closing the NLP loop of model development [58]. In the context of scoring written summaries, engineers grapple with two difficult tasks when interpreting transformer-based language models specifically. The first is *understanding how variations in model inputs can lead to changes in model outputs* **(T8)**. For example, counterfactual explanations [19, 55, 65] are often used to explore model behavior and robustness by systematically varying and comparing different types of revisions. The second is *interpreting the internal model weights (i.e. attention in transformers)* **(T9)** to understand what the model is capturing about summaries. This could enable engineers to identify emergent representations of semantic knowledge and lexical structure that contribute to certain model scores [42]; e.g., identifying salient interactions between specific character spans across model layers and heads. Providing both external and internal model probing methods can guide users in discovering useful insights to help them close the NLP loop of model development.

C4 Bridging global and local model behavior. In addition to existing challenges interpreting deep learning model behavior, learning engineers also face a scale issue, as the models used in *iTELL* can score textbook sources and summaries thousands of words (tokens) in length. This makes summarizing the sheer scale of tokens and attentions while helping users make sense of the data difficult. To address this, engineers seek methods to *summarize the interactions between all tokens in the same interface* **(T10)**. Bridging the gap between global and local model behavior can be supported by *comparing interactions between tokens, layers and heads* **(T11)** while enabling users to *drill down to specific token interactions at specific layers and heads* **(T12)**. By aggregating data at multiple levels, our system should reveal subsets of interesting interactions that help engineers make sense of complex model behaviors.

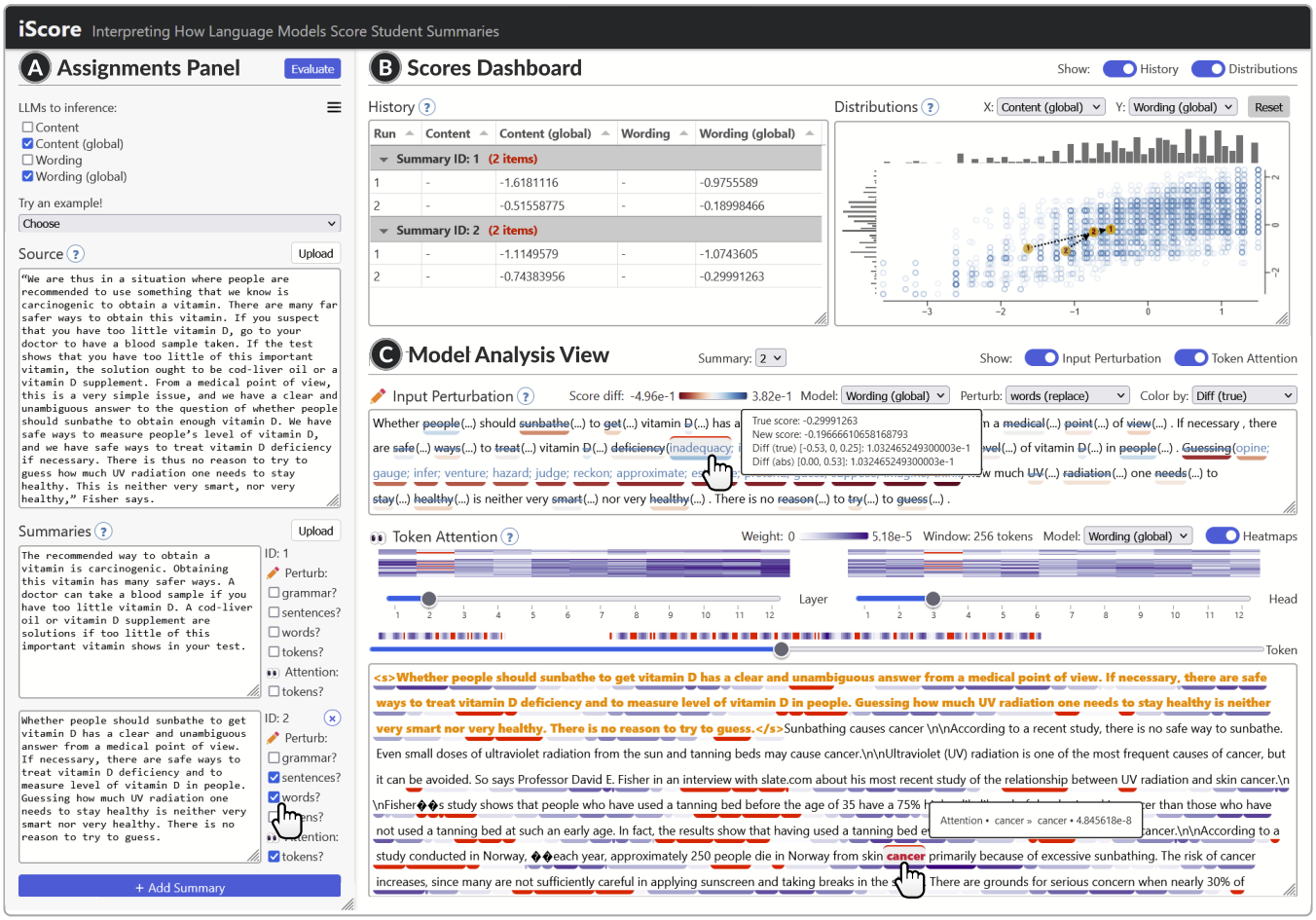

Figure 2: *iScore* visualizes multiple LLM-scored writing samples to help learning engineers interpret model performance. Above, a learning engineer interprets how two plagiarized summaries are scored across two runs (Sect. 5). Users can upload, score and manually revise and re-score multiple source/summary pairs simultaneously in the *Assignments Panel* (A), visually track how scores change across revisions in the context of expert-scored LLM training data in the *Scores Dashboard* (B), and compare model weights between words across model layers/heads, as well as differences in scores between automatically revised summary perturbations, using two model interpretability methods in the *Model Analysis View* (C).

4 THE *iScore* SYSTEM

Based on our design challenges and user tasks, we developed *iScore*, an interactive visual analytics tool for learning engineers to upload, evaluate, and visualize the results of automatically scoring summary writing using LLMs. Our system helps users characterize quality in writing samples and enable transparency in LLM performance by tightly integrating three coordinated views. In the *Assignments Panel* (Sect. 4.1), users scaffold the evaluation process around comparing multiple source/summary pairs simultaneously. After evaluating the pairs, the *Scores Dashboard* (Sect. 4.2) visualizes the provenance of scores in context of LLM training data, while the *Model Analysis View* (Sect. 4.3) presents two different LLM interpretability methods at multiple levels of abstraction.

4.1 Assignments Panel

The *Assignments Panel* (Fig. 2A) provides multiple source/summary text inputs and model analysis options. This structure guides users to compare different facets of summary writing in the *Scores Dashboard* and *Model Analysis View* by choosing between multiple model analysis methods.

We scaffold the evaluation of source/summary pairs in several ways. Users can choose any number of language models they have trained from a list to customize how they evaluate assignments and which models are being compared **(T1)**. We also provide a list of example source/summary pairs to help users understand how to start inputting sources and summaries. Users can then provide a single source, either through copy/paste, upload, or by directly typing and editing in a free text box. For the given source, users can add any number of summary text boxes with the same input capabilities **(T2)**, with a list of model analysis option checkboxes

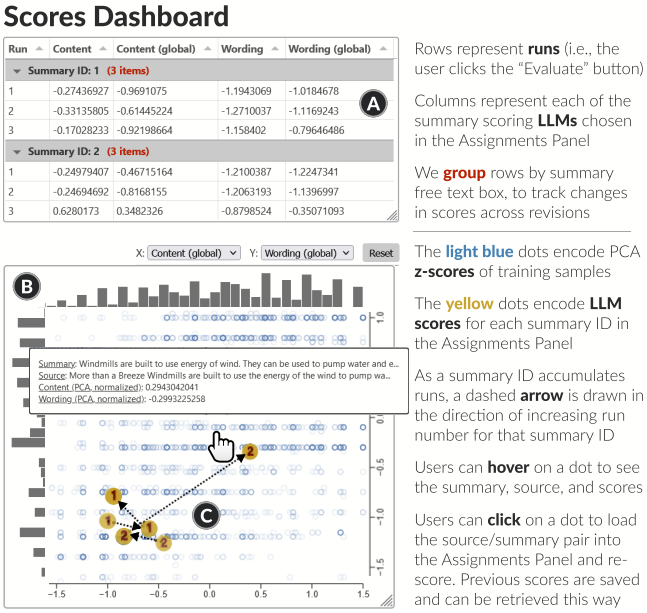

Figure 3: Breakdown of the *Scores Dashboard*. The table and scatter plot help users compare variations on summaries by tracking how scores change across manual revisions.

for each that will populate the visualizations in the *Model Analysis View* for each summary with a checkbox selected **(T8, T9)**. Because each model analysis option can be computationally expensive, users can toggle which options they would like to run independently for each summary. We explore these options in detail in Sect. 4.3.

Source/summary pairs are scored in real time using an API that interfaces with a Python Flask server running PyTorch implementations of the language models. We use the HuggingFace Transformers [62] API to load models and perform sequence classification by combining each summary with the source using a separator token. Users can rapidly generate multiple test cases across several summary variations for a given source and evaluate them simultaneously, saving time and improving efficiency.

4.2 Scores Dashboard

After scoring the source/summary pairs, we provide an overview of model scores for each summary in the *Scores Dashboard* (Fig. 2B). We visualize the provenance of score data to help users gain insight into how changes in summaries affect model scores across runs of the model at different levels of aggregation.

Each time the user scores a set of source/summary pairs, e.g., when revising a summary to see how scores change, we visualize the provenance of raw summary scores as a table (Fig. 3A) **(T6)**. Runs of the models are numbered. Rows represent a single source/summary pair, labeled by run and grouped by summary free text box. Columns display the scores for each LLM chosen in the *Assignments Panel*. In this way, we facilitate rapid comparison across both runs for a single summary, as well as across summaries for the same run. For example, users can easily determine if scores are increasing/decreasing when a single summary is revised or discover which summaries are performing better compared with others across runs. Users may also want to understand how summary scores from the *Assignments Panel* compare with the ground truth model training examples (Sect. 3.1.2), helping them gain additional context into why a particular summary received a certain score. To facilitate this, we visualize training example scores directly as a scatter plot with overlapping blue dots (Fig. 3B) **(T3)**. We then plot the scores of the current summaries in the *Assignments Panel* as yellow dots overlaid on top of the training example score distributions (Fig. 3C). To dig even deeper, our collaborators requested the ability to re-score the training examples, helping them better understand the differences between training and testing on the ground truth data set. In response, we allow users to click on any dot in the scatter plot to load that source/summary example into the *Assignments Panel* and score it. Scores are saved and can be loaded on the fly by clicking on the corresponding dot **(T7)**.

As users explore the history and distributions of scores, we provide several interactions and useful overlays. To help users gain additional context and insight, we visualize the provenance of summary scores in the *Assignments Panel* by plotting each run of a summary as a dot numbered by the run **(T6)**. A dashed arrow line indicates the direction that the score is moving for that summary between runs. To solve issues with occlusion, we made each training example dot mostly transparent, revealing patterns in training example score distribution density where dots overlap. We also explicitly plot the distributions of training example scores as bar chart histograms along the scatter plot axes **(T3)**. To see details on demand, users can hover on dots to reveal a snippet of the source/summary pair text for that example as well as the numeric scores. They can also hover on bars to view the bin size and range.

4.3 Model Analysis View

The *Model Analysis View* (Fig. 2C) allows users to interpret and gain deeper insights into how the LLMs are scoring each source/summary pair in the *Assignments Panel*. We do this by leveraging two broadly effective techniques for interpreting deep learning models at multiple levels of text aggregation: (1) visualizing the results of input perturbation on model outputs in the *Input Perturbation* plot (Sect. 4.3.1); and (2) visualizing attentions of transformer-based language models between tokens across every model layer and head in the *Token Attention* plot (Sect. 4.3.2). We visualize these methods applied to a single source/summary pair at a time; users can switch between pairs using a drop-down menu.

4.3.1 Input Perturbation. Perturbing input parameters during model inference has been widely used to explore the inner workings of black-box AI models, particularly in scenarios where complex model architectures make it difficult to trace explainability through the entire model from input to output [33, 41]. In the case of using transformer-based language models, we created four summary text perturbation options in the *Assignments Panel* **(T5)**. The **grammar** and **words** perturbation options both replace word-level spans by fixing the spelling and using synonyms respectively, while the **sentences** and **tokens** perturbation options mask sentence-level and token-level spans from the summary respectively. After automatically perturbing each summary, we inference each LLM requested in the *Assignments Panel* and visually compare the new summaries'

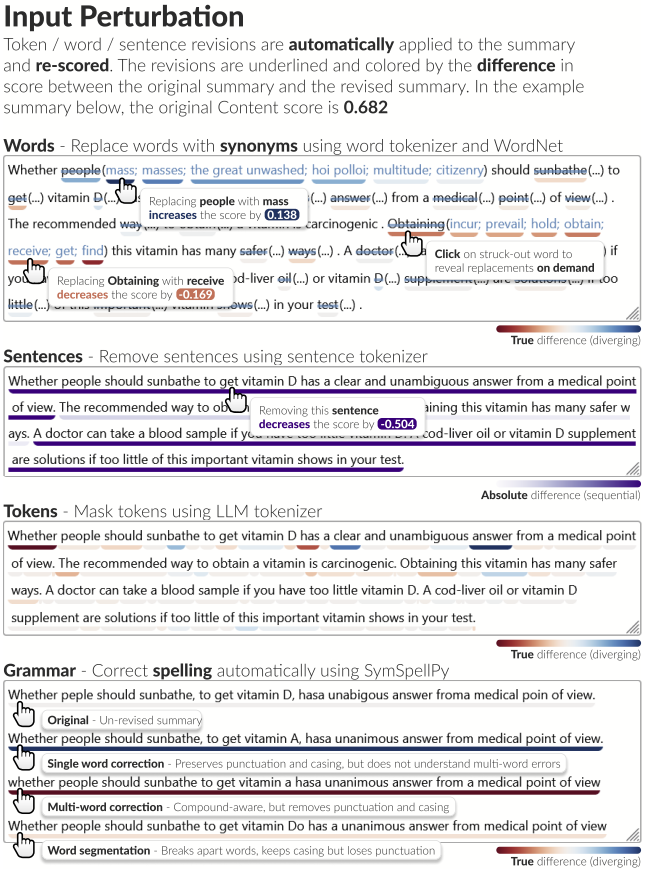

Figure 4: Breakdown of the *Input Perturbation* visualization. Multiple perturbation methods help users test hundreds of different kinds of revisions at scale by automatically applying and re-scoring summaries for them.

scores with the original, unperturbed summary score **(T4)**. These perturbation options mirror typical methods of evaluating summary writing suggested by our domain expert collaborators, and were developed and tested as a proof of concept.

Data. The **words** option replaces all word-level spans that are not English stop words or punctuation with synonyms in each summary, giving users an overview of which words have a strong effect on each summary's score when replaced and if semantically similar word replacements could improve the score. We perform replacement by first identifying all word-level spans using a treebank tokenizer, then looking up the synonyms of each word and returning each synonym's lemma. Words are replaced by their synonyms' lemmas and a new summary score is computed for each lemma, resulting in a list of scores for each synonym replacement.

Both the **sentences** and **tokens** options apply masking to the summary at the sentence-level and token-level, respectively. Masking spans attempts to replicate saliency in model interpretability, which assigns a score to each input based on the strength of its contribution to the final output. In *iScore*, we use the difference in score when masking spans as a proxy for the importance of the span to the unperturbed summary score. We use a sentence boundary detection tokenizer to locate each sentence-level span in the summary, while we use each LLM's tokenizer used for inference to extract token-level spans that correspond with the tokens in the *Token Attention* plot (Sect. 4.3.2). As with the **words** option, we compute a new summary score for each masked span.

Finally, the **grammar** option attempts to automatically correct the spelling of each summary. This allows users to quickly explore how the quality of the grammar affects each summary score. We first perform a correction on the entire summary then query each model with the corrected summary, to get a single score. We implemented three correction variations for comparison. The first performs single word spelling correction using lookups on individual word spans generated from a treebank tokenizer. This allows us to accurately preserve the structure of the sentence (i.e. the casing and punctuation) but may miss context-dependent spelling suggestions that are likely to occur. The second performs compound-aware multi-word spelling correction on the entire summary. In contrast with single word spelling correction, the accuracy of recognizing and fixing multi-word errors is likely to be higher yet the structure of the sentence is lost in the process. The third attempts word segmentation to divide multi-character spans into their constituent words, preserving casing but ignoring punctuation.

Visualizations. To visualize the results of perturbation, we plot the entire summary text in a text box and underline text spans (token-, word-, or sentence-level) that correspond with the perturbation options chosen in the *Assignments Panel* (Fig. 4). We then encode the difference between each perturbed summary's score and the unperturbed summary's score on a color ramp and apply the color to the text underlines **(T4)**. This encoding scheme allows users to quickly identify and compare the length and effect size of each span in the perturbed summary with the unperturbed summary score. Users can choose between a red-to-blue diverging color ramp representing the true negative-to-positive difference in scores, or a transparent-to-purple linear color ramp representing the absolute value of the difference in scores. We display a legend for the underline colors above the plot. To get details on demand, users can hover on an underlined span at any level to get the exact difference values. Finally, we provide drop-down menus to let users switch between models, perturbation options, and color ramps.

The summary text and underlines are plotted differently based on the perturbation option (Fig. 4) **(T5)**. For the **words** option, we implemented a word replacement view common in most text editors with markup capabilities. Replaced words have a strike-through decoration and are followed by ellipses, denoting a replacement was made for that word. This gives users an overview at a glance of which words have hidden synonym replacements that affect the unperturbed summary score the most. Users can click on any word with replacements to show/hide that word's synonyms on demand. Each synonym is shown in a list with a text underline colored by the difference in summary score when using that synonym in place of the original word. The replaced word is also underlined; the underline is colored by the maximum signed magnitude of score differences for all synonym replacements of that word. For the **sentences** and **tokens** options, we underline each sentence-level and token-level span respectively with the color representing the

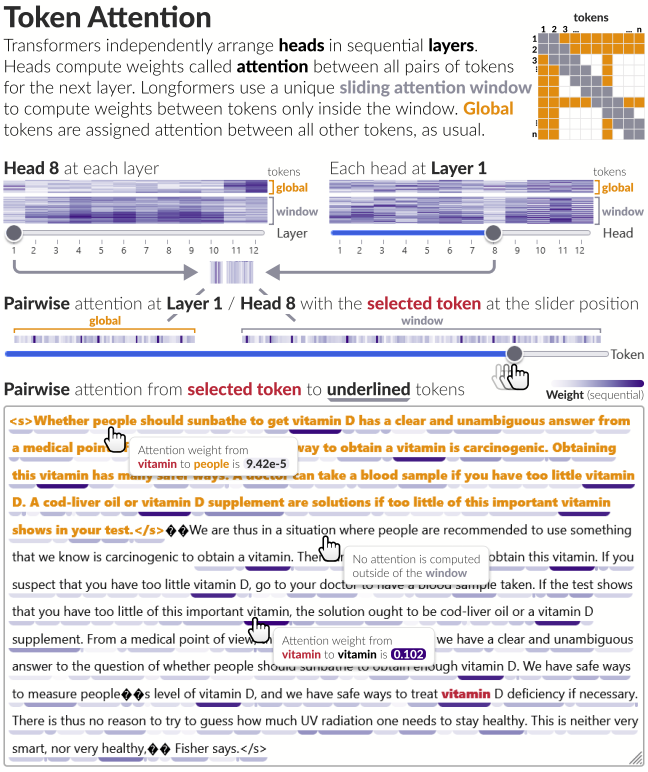

Figure 5: Breakdown of the *Token Attention* visualization. The combination of heat maps, rug plot and text underlining helps users make sense of complex model behaviors by keeping them in the loop at multiple levels of abstraction.

difference in summary score when that span was masked during inference. This allows users to quickly identify spans with strong effects on the final model output. For the **grammar** option, we plot and underline the entirety of each correction variation of the unperturbed summary in a single color representing the difference in the corrected summary's score. Users can then compare changes in grammar and scores side-by-side between corrections.

4.3.2 Token Attention. Complementing our approach of evaluating models externally, we also visualize the internal multi-headed attention mechanisms underlying our transformer-based language models. By visually connecting attention to the semantic and syntactic structure of our source/summary pairs, we aim to reveal patterns and associations in the text that are strongly contributing to the final model output. This can help users gain deeper insights into how different training procedures and training data lead to prioritizing different spans between layers/heads, and compare how attentions and these priorities change between models.

Data. Prior work [54, 59] has successfully explored using graph and matrix visualization techniques for visualizing pairwise attention between tokens for models with smaller sequence lengths (< 256 tokens). However, a major challenge in scaling attention visualization for our models is the size of the sliding context window (256 or 512 tokens) compared to the input sequence length (up to 4096 tokens) for Longformers [2]. Longformer assigns **global attention** to one or more tokens and a **sliding attention window** which moves across the text. Tokens in global attention generate attention to every other token, while tokens in the sliding attention window only generate attention to other tokens within the window. This creates issues in visualizing the sheer number of tokens and attentions both locally (within the context window) and globally (for all tokens), while helping users make sense of the data. For example, using a 256-token sliding context window around each token, a source/summary pair with 700 tokens (≈ 500 words, such as a 5-paragraph source and 1-paragraph summary) results in a 4-dimensional matrix of $700 \times 256 \times 12 \times 12 \approx 26$ million attentions between each token for a model with 12 layers and 12 heads in each layer. If we consider global attention between every token, this number increases to $700 \times 700 \times 12 \times 12 \approx 70$ million attentions. Textbook sources and summaries in *iTELL* regularly exceed 4000 tokens (> 150 million attentions). Because our models inference using both the source and summary, we considered designs that allowed us to scale up to 4096 token attentions visualized at once.

Visualizations. To help users discover higher-level patterns and drill down into specific layers and heads, we provide an overview of attention from the selected token to all other tokens using heat maps (Fig. 5) **(T10)**. We plot two different heat maps as uniform grids of all other tokens around the selected token arranged vertically, and each layer (i.e. the Layer heat map) or head (i.e. the Head heat map) arranged horizontally. Cells are colored by the attention weight from the selected token to the other token (row) at that layer or head (column). A horizontal single-ended range slider is aligned below each heat map and allows users to select a particular layer/head combination for viewing in the text box described below. As the range slider for the Layer or Head heat map is dragged, the other heat map not being dragged will update. For example, if Layer 2 is selected in the Layer heat map, then the Head heat map will show an overview of attentions across all the heads of Layer 2. This allows users to quickly pan through layers and heads to gain insights into how attention changes in other heads and layers, respectively.

Below the heat maps, we allow the user to drill down and see all pairwise attentions between a selected token and all other tokens at the current layer/head selected in the heat maps (Fig. 5) **(T11)**. To do this, we position a rug plot above a horizontal single-ended range slider that represents the index of the selected token, and visualize the attention weight from the selected token to all other tokens at each index. Similar to the *Input Perturbation* plot, we then plot both the source and summary text in a text box and underline each token **(T12)**. Each token underline is colored by the attention weight from the selected token to that token. In this way, we address issues of scale as the number of tokens increases by leveraging basic text formatting to help users read the text naturally while comparing attention weights at the same time. Users can manually click on any token in the text box to update the heat maps, rug plot, and text underlines with updated attention weights from the newly selected token. They can also use the range slider to quickly pan through tokens and see patterns at the dataset-level.

We provide several interactions and visual embellishments. A single linear transparent-to-purple color ramp is used to represent attention weight for the heat maps, rug plot, and text underlines.

Where attention is zero (not missing) between tokens, we color cells, strips, and underlines in bright red to make these outliers visually distinct. We display a legend for the heat maps above the plot. Because our models use a sliding context window, in all plots the attention weights for tokens outside of the context window are not plotted, and attentions from the selected token to all global tokens are always shown. In the text box, global tokens have a golden stroke applied and the selected token has a red stroke applied, to visually distinguish them from other tokens. To get details on demand, users can hover over any cell, strip, or token and get the token text and exact attention weight value. Finally, we provide drop-down menus to let users switch between models.

4.4 Implementation

iScore is an open-source[2] web app built using D3.js [5]. We implemented our input perturbation methods in Python using symspellpy[3] for revising grammar and NLTK's [4] WordNet interface for word synonym replacement and sentence masking. *iScore* uses HuggingFace Transformers API [62] to load Longformer pre-trained models and process all source/summary pairs in real time.

5 USAGE SCENARIO

A learning engineer is manually revising two plagiarized summaries of a textbook source in *iTELL* and comparing their LLM-assigned scores using *iScore*. They seek to: (1) see how they can trick the models into giving the summaries higher scores by reordering and rephrasing sentences; and (2) determine if modifying or removing certain phrases such as key words and main ideas will affect model scores. They hope to use their findings to develop feedback for learners writing summaries of textbook sections on how to improve scores, as well as generate additional training examples that ensure the models are resilient to learners trying to "game" the system.

They start by using the *Assignments Panel* to upload a source text and two summaries **(T2)**. While they expect that the models will score plagiarized summaries low on both Content and Wording, they are unsure how resilient the models are to reordering and rephrasing, and whether changes in Content and Wording scores will differ **(T1)**. In the first run, they score the plagiarized text verbatim, resulting in negative scores that indicate the summaries are below average. In the second run, they reorder the sentences without modifying words. The history table in the *Scores Dashboard* shows a slight increase in both scores between runs, yet the scores are still negative, a good sign **(T6)**. They try a third revision, by paraphrasing each sentence without changing the order of sentences. Interestingly, the dashed arrow lines in the distributions scatter plot show a stronger increase in Content over Wording. Finally, they reorder and paraphrase the summaries to test the combined effects. This time, Content increases further but Wording actually decreases. If paraphrasing alone improves both Content and Wording while reordering and paraphrasing only improves Content, the learning engineer worries that learners may be able to paraphrase summaries using paraphrasing tools like ChatGPT that could pass. To verify this pattern, the engineer clicks on a training source/summary pair with similar scores in the distributions scatter plot to load it in the *Assignments Panel* **(T3, T7)**. They apply the same reordering and rephrasing, and find it produces a similar score pattern. By understanding the effects of reordering versus rephrasing on model performance using *iScore*, the learning engineer can augment training samples and re-train new model versions. Once they trust the re-trained models are robust, they can confidently warn learners that the models are aware of these methods and that summaries should be written from scratch else they will fail.

The learning engineer wants to follow up on their intuition that certain words being rephrased or sentences being removed leads to changes in score. However, it would be time-consuming and inefficient to manually replace every word and remove every sentence, and certain replacements could be missed. Using the latest revised versions of the summaries, they choose the sentences and words perturbation model analysis options in the *Assignments Panel* for each summary **(T5, T8)**. Viewing the sentences option in the *Input Perturbation*, they are surprised to see that the first sentence tends to drop the score dramatically when removed **(T4)**. Our collaborators expand on the potential reason for this result in the expert feedback in Sect. 6.2. Switching to the words option, the engineer finds that more complex synonyms generally increase the Content score, indicating a potential learned association between word complexity and score. For the purposes of the summary assignment, this is considered a positive learned association. However, they then check the Wording model and find the same pattern does not hold; synonyms not found in the source tend to lower the Wording score. This is surprising, since Wording consists of rewarding summaries that go beyond the source text. The engineer makes two decisions: (1) re-train the models by exposing them to more complex summaries; and (2) give learners feedback that they should generally use a variety of vocabulary that goes beyond the source text, but that does not change the meaning of their sentences.

6 EVALUATION

Our goal for evaluation was to validate the effectiveness of *iScore* for helping learning engineers understand, evaluate, and build trust in how their large language models (LLMs) automatically score summary writing. To do this, we first deployed *iScore* with three of the learning engineers (domain experts) with whom we collaboratively designed *iScore* over the course of a month. We first describe a case study detailing how *iScore* enabled one of the engineers on our team to improve the accuracy of LLMs used in *iTELL* by three percentage points (Sect. 6.1). We highlight which challenges and tasks raised in Sect. 3.2 and Table 1 they achieved. We then conducted individual semi-structured interviews ($1-2$ hours long each) with all three engineers on our team and collected feedback [43] on the ways *iScore* generally impacted how they understood, evaluated, and trusted the LLMs used in *iTELL* (Sect. 6.2).

6.1 Case Study

One of the domain experts on our team that we deployed *iScore* with and interviewed is a learning engineer who has expertise in natural language processing (NLP). He is fairly new to deep learning and wants to improve the LLMs deployed in *iTELL* for automatically scoring summary writing. To do this, he trains several LLMs using various combinations of parameters and evaluates their

[2] *iScore* models and code: https://github.com/AdamCoscia/iScore
[3] symspellpy code: https://github.com/mammothb/symspellpy

performance on sample writing by comparing scores one at a time. He is interested in "opening the black box" of the LLMs and feels the most important use of *iScore* is helping engineers confirm that their model is doing what they want it to do: *"You don't see the model working. You set everything up as carefully as you can but it is all done in the GPU. It's easy to make a mistake, allow information leaks, and there is nothing to 'see'. You don't know where you might have messed up."*

During deployment, he used the *Token Attention* visualization to investigate the relationship between the summary and source text. Looking at the Content model, he noticed that the attention weights tended to converge in the final model layers to high values only in a small radius around a selected token (i.e., darker purple underlines fading linearly to light purple for tokens farther away) **(T9, T10)**. This did not match his mental model of what the LLM should be doing. Tokens in the source should pay attention to the entire summary in each layer, even when they are far away from the summary. He wondered whether the LLM was using all available information to determine the final score. He quickly compared this qualitative pattern with other source/summary pairs as well as with the Wording model using the *Assignments Panel* and confirmed the same insight. He realized that, *"by default, only the classification token receives attention. By using the [token slider in the] attention visualization, I watched the attention window slide across the text. It was obvious that only one token was getting global attention."* **(T11, T12)** The insight gave him an idea to improve the model, by assigning global attention to the entire summary during training and inference. After retraining, he found the Content score improved significantly, from $r^2 = 0.79$ to $r^2 = 0.82$. He reflected on this improvement, saying *"It's a simple thing, but the attention visualization confirms that the model now does what the model was supposed to do! I gained an important insight using iScore that changed the model. It was actionable information that led to a better model."*

Overall, the learning engineer overcame the challenges outlined in Sect. 3.2 by using *iScore* to draw a relationship between summaries and scores **(C1)**, uncover discrepancies between models and differently scored summaries **(C2)**, deeply investigate errant model parameters **(C3)**, and fluidly interact with the visualizations to transition between global and local model behavior **(C4)**.

6.2 Expert Feedback

Our three domain experts were a team of graduate students broadly interested in computational work at the intersection of linguistics, learning sciences and NLP. These experts were the same collaborators involved in the design process and carried out the deployment. Some of the experts contributed to the development of the *iTELL* framework, and all had experience working with *iTELL*. All studied or taught language and NLP in various ways, including work in linguistics, language learning, development, and teaching, and using LLMs to predict text readability, reading comprehension, personally identifiable information, and build other learning tools. All three experts self-identified as male. We organized the interview questions and domain experts' feedback around three broad questions we directly asked each collaborator. We then conducted inductive thematic analysis [7] of the feedback to identify emergent themes that were discussed amongst all authors.

(1) **Understanding LLMs** – "How did *iScore* improve your understanding of LLM performance?"
(2) **Evaluating LLMs** – "How were the visualizations and interactions in *iScore* useful for interpreting LLMs?"
(3) **Transparency and Trust in LLMs** – "How did *iScore* enable transparency and promote trust in using LLMs?"

(1) Understanding LLMs. We first inquired about the major challenges the experts face in understanding LLMs. We found several recurring themes: understanding how decisions are made from billions of LLM parameters; investigating whether LLMs use context around words; and "seeing" what is going on inside models using traditional programming tools. There were also architecture-specific challenges related to Longformers, as one expert explained: *"Since we are using Longformer with the sliding attention window, there are developer degrees of freedom that can be tweaked compared with other models."* These were important to address for several reasons. One expert was concerned with the long-term reproducibility of their findings, feeling that *"there is always anxiety or tension around whether an outcome is just a fluke."* Another described how addressing these challenges can also lead to new research questions: *"Sometimes we couldn't explain [findings] with the visualizations, but it led us to ask new questions about our research, and it served to remind us that we don't always know how [the LLMs] are working."*

iScore enabled the experts to learn several new things about their models. Using the *Token Attention* visualization, one expert discovered that punctuation tended to contain a lot of information in final model layers, as the marks were always underlined in dark purple. Consequently, removing punctuation would cause scores for both Content and Wording models to go down drastically. They were concerned with the ramifications of this finding: *"We didn't expect to see attention weights on punctuation... The question is, does this have any pedagogical impact?"* Another expert found that removing the first sentence from summaries would drop scores by nearly 90%. They hypothesized that the models were focusing on the beginning of the sentence and the punctuation denoting the end of the first sentence, as the classification token is at the start of the input. This may corroborate the findings from our case study, as the retrained models with global attention on the entire summary did not exhibit this behavior as strongly. The expert reflected on this behavior, saying *"While [topic sentences] could be important for writing summaries, it may not be the way the model is operating."*

(2) Evaluating LLMs. To discover these insights, the experts felt the *Input Perturbation* and *Token Attention* visualizations allowed them to see how things worked at a higher level. One expert summarized how this helped them, saying *"we now have methods to help us show that the models work as we expect, and also when we don't expect results."* For example, the same expert described how they used the *Assignments Panel* and *Input Perturbation* visualizations to create summary variations and analyze their models: *"I was using input perturbations to illustrate how adversarial attacks can trick models. This led me to new research questions. The perturbations are one word at a time. What if we do n-gram changes?"* Yet there was room for improvement, as another expert sometimes had trouble knowing what to look at in the *Token Attention* visualization: *"There seems to be a consistent noise in the attention. What is significant over*

and above the noise?" In the future, *iScore* could include statistical analysis such as highlighting attentions far outside of the mean.

As they explained their process for discovering insights with *iScore*, two of the experts developed new ideas for how the visualizations could help them close the loop of model development. One used the *Model Analysis View* to quickly profile and decide between the best-performing models: *"I often train multiple models for a task and want to compare them. What I like to do is switch between the models using the drop-downs."* To promote reproducibility in the future and communicate their findings directly in the *iScore* interface, they requested hypothesis testing capabilities such as a chi-squared test. The other expressed a desire to use the *Scores Dashboard* for tracking changes across a specific task, such as key phrase analysis: *"It may be too easy to just throw in the key phrases and increase model scores and we don't want that."* This type of analysis could generate additional examples for injecting into the LLM training data, to make them robust to learners trying to "game" the system by spamming the input with key phrases.

(3) Transparency and Trust in LLMs. The experts all reported different factors and requirements for building trust. Reproducibility was a requirement for one learning engineer, as they wanted to avoid unexpected surprises especially when deploying their LLMs: *"You might get great results, and sometimes they look too good and you have to go back and check your code."* Addressing biases primarily in the training data was mentioned by another, though the expert also noted the difficult endeavor of explaining the vast number of LLM parameters responsible for producing a score as a barrier to realizing transparency. Another expert believed in clearly demonstrating model limitations and edge cases, especially given the ability for LLMs to hallucinate: *"If we are dead set on fooling models, we can create examples to fool the models."* Finally, the context of deployment was considered important for all experts, as one expert explained: *"Convincing stakeholders becomes easier if you can show them examples of how the model is working."*

The experts unanimously agreed that the most useful feature of *iScore* for inspiring trust was the *Input Perturbation* visualization. One expert felt that, *"in a way, input perturbation allows you to put in adversarial examples to test your trust in the model."* At the same time, another expert commented that the *Token Attention* visualization did not achieve complete model transparency for them. Following up, they still expressed optimism that *"trust can be achieved even without full transparency"*, especially when using the *Assignments Panel* and *Input Perturbation* visualization: *"The aspects of iScore that can test variations allow us to demonstrate what happens with changes in summaries. This can broadly allow us to improve trust in these systems."* This benefit extended outside of the development environment, allowing the experts to show their work in context of deployment with stakeholders. On the opposite end of the spectrum, for developers deep in the NLP loop, one expert felt *iScore* provided a layer between development and production where few tools exist to provide qualitative feedback to learning engineers at the intersection of educational data and machine learning. For them, *iScore* promoted trust through reproducibility because the parameters of each LLM can be quickly iterated, tested, and reported using screenshots of the system.

7 DISCUSSION

7.1 Implications For Design

By distilling the findings from our evaluation, we synthesized design implications for future visual analytics systems situated in terms of our key design challenges **(C)** outlined in Sect. 3.2 as well as prior literature.

Structure LLM evaluation using visual hierarchies. We observed that communicating the LLM evaluation process through visual hierarchies enabled thoughtful comparison between LLM inputs and outputs, improved understanding of how models work and increased trust that models are working correctly. To scaffold comparison, users should be able to easily upload and/or write multiple text inputs to their LLMs at once **(C1)**. Text inputs should be arranged to visually communicate comparison (e.g., grouping by topic, version, author, etc.) and include options for automatically testing multiple models simultaneously **(C1)**, saving users time and aiding their sensemaking process. Further, we found that visualizing the distribution of LLM outputs across groups of models, inputs and revisions helped users communicate valuable insights when reporting their findings. One way to enrich visual comparison is visualizing distributions in context of the "ground truth" model training data **(C1)**. Another way is to give users options for grouping and visualizing multiple types of automatic text revisions **(C2)**. Finally, our users suggested visually highlighting and grouping potential relationships between all tokens using statistical tests **(C3, C4)**. We discuss future work in this area in Sect. 7.4.

Scale LLM interpretability methods to large inputs. Supporting analysis of multiple large text inputs became both a computational and mental bottleneck, highlighting the challenge of scaling interpretability methods as LLMs are developed that can process larger inputs [63]. Leveraging fluid interaction [15] can help users identify and track sets of tokens that a model is focusing on and compare these sets against other tokens at multiple levels of aggregation. For example, users should be able to visualize and compare differences between text inputs at multiple levels of aggregation (e.g., token, word, sentence) as well as interactively switch between levels on the fly **(C2, C3)**. Interactively specifying encodings for how changes to text spans affect LLM outputs using inline annotations (e.g., underlining tokens, words, sentences, etc.) allows users to read text naturally at scale while fluidly identifying patterns between individual and groups of annotations, even as inputs grow. We found this to be useful both when externally probing models such as input perturbation and when analyzing internal model weights **(C3)**. As interpretability methods can be computationally expensive for large numbers of tokens, users should also be able to subset and apply methods both simultaneously and independently to any number of inputs **(C1)**. Result sets should then be easy to switch between on the fly, allowing users to quickly visualize and compare differences without slowing down the user experience **(C1, C3)**.

Potential future applications. Taken together, these design implications inspire future applications of visual analytics for interpreting LLMs. For example, consider an engineer training LLMs for a question-answer module, or a NLP researcher investigating LLMs trained to assign a readability score. They could import their models into *iScore* to interpret what parts of the text the models are

paying attention to using the *Token Attention* visualization, as well as how the models respond to different perturbations of the input text using the *Input Perturbation* visualization. Developers could also deploy the perturbation features with users to collect feedback for retraining their models, by analyzing how inputs are revised and how scores change as users respond to seeing model scores.

7.2 Responsible and Ethical AI for Education

The scalability and generalizability of LLMs could enable summary writing assignments to be used in a wider variety of environments, including Massive Open Online Courses (MOOCs), learning platforms and intelligent textbooks [18]. To realize this, there are several broad practical and ethical considerations of using machine learning (ML) and artificial intelligence (AI) that *iScore* addresses, helping learning engineers deploy responsible AI in educational tools like *iTELL* and mitigate the pedagogical risks of using AI-powered tools in critical learning environments [27].

It is necessary to remove potential biases and evaluate the fairness of any AI model's predictions. Ensuring fairness means using balanced data sets during the training period as well as oversight and regular re-training based on the performance of AI models in production. Using *iScore*, learning engineers can rapidly explore and identify potential biases that manifest either internally in the model weights or externally in the model scores. For example, they could track patterns of biases in score assignments across revisions containing different personal identity phrases such as pronouns. AI-powered tools for education should also accommodate the target language of learners by providing both equitable and interpretable outputs. This can be achieved by expanding LLM training across a variety of data sets in various languages, and using the input and history tracking features of *iScore* to evaluate LLM performance at scale across many different summaries. Finally, perceptions of AI are an important factor for educators in the adoption of AI-powered education technologies, with trust being one of the most important determinants of whether an AI model is actually used [44]. One of the greatest barriers to building trust is a lack of technical expertise among educators in understanding what AI is capable of. We believe *iScore* contributes practical methods for enabling transparency and helping non-ML-experts build trust in deploying AI models like LLMs in tools like *iTELL*. For example, the perturbation features of *iScore* could be adapted and simplified for presentation to non-ML-expert audiences, such as instructors and learners, as a hands-on version that seeks to promote trust in using AI-powered educational technology. In the fields of education, machine learning and HCI broadly, it is imperative to develop tools like *iScore* that enable engineers to build ethical and responsible AI tools and interpret them well enough to begin building the essential trust necessary to see them implemented safely in learning environments.

7.3 LLM Generalizability

In this paper, we demonstrated the capabilities of *iScore* using Longformer, a pre-trained LLM based on RoBERTa, as Longformer allows for longer text inputs that combine summary writing and source text during inference. RoBERTa itself is a fundamental language model that continues to obtain state-of-the-art results on classification and natural language understanding tasks [46]. Beyond RoBERTa, *iScore* can be used to interpret any transformer-based model that utilizes a separator token for sequence classification, such as a reference text and a summary or a question and an answer, building on the successes of other transformer-based LLM architectures. This includes both encoder- and decoder-only models as well as encoder-decoder models such as GPT-3 [8] and T5 [39]. To enable larger input text sizes, StreamingLLM [63] is a recent framework that can replicate the larger input text size capabilities of Longformers using attention sinks for models such as LLaMa [51]. Finally, to make our methods accessible to learning engineers and NLP researchers in the future, we use transformers downloaded from the HuggingFace Transformers [62] model library that are open-source and lightweight enough to run on a single GPU.

7.4 Limitations and Future Work

Several analytical limitations were expressed during the interviews. While our collaborators were positive about moving towards using visualizations that show model weights, they still had issues with interpreting them. One felt that there was a constant noise in the perturbation scores and attention weights that made it difficult to identify important patterns and relationships. Interactions at specific layers/heads were also difficult to explain, as one engineer disliked scanning across layers/heads to perform an "axiomatic" type of analysis. While attention visualizations enabled our collaborators to improve model understanding and accuracy in Sect. 6.1, there is on-going debate whether attention weights in transformers can be used as a reliable source of interpretation for model performance [1, 11, 26, 61]. Several engineers suggested augmenting *iScore* with statistical tests that could improve how to interpret and compare attentions and perturbations. These could include hypothesis tests such as a chi-squared test to compare distributions of attention weights between models or a score outlier detection method using interquartile ranges. Further, our collaborators found the input perturbation methods to be the most useful, yet this was also the most computationally demanding method, which can limit the responsiveness and increase deployment requirements for this feature. One promising direction to emulate perturbation is performing gradient-based attribution of input features to the final model prediction [49], by developing new methods specifically for the sliding attention window in Longformer models. Perturbation also only considers the marginal contribution of a token/sentence among the remaining ones, ignoring interactions among features (i.e., token/sentence). Using Shapley Values (e.g., SHAP [33]) may be a more accurate approach, albeit at the cost of increased computational time over the already expensive perturbations. Finally, one collaborator requested a method for performing automatic sentence-level replacement, to investigate how this could improve writing feedback in the context of automatic scoring. This could be achieved by using additional LLMs to generate grammatically and/or semantically similar sentences for replacement.

8 CONCLUSION

As educational technologies increasingly utilize "black box" deep learning models, it is critical to empower model developers to understand, evaluate, and build trust in their models before deploying them in critical learning environments. In this work, we conducted

a user-centered design process with a team of learning engineers who train and deploy large language models (LLMs) that automatically score written summaries of source texts. We presented *iScore*, a visual analytics system for learning engineers to upload, compare and visualize multiple LLM-scored summaries. By structuring evaluation of LLM performance around comparison of multiple source/summary pairs, visualizing the provenance of summary scores in context of the LLM training data, and presenting different LLM interpretability methods simultaneously at multiple scales, *iScore* increases LLM transparency and trust throughout human-in-the-loop evaluation of LLM performance.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 2247790 and Grant No. 2112532. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. 2020. A Diagnostic Study of Explainability Techniques for Text Classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 3256–3274. https://doi.org/10.18653/v1/2020.emnlp-main.263

[2] Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150* (2020).

[3] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (Virtual Event, Canada) *(FAccT '21)*. Association for Computing Machinery, New York, NY, USA, 610–623. https://doi.org/10.1145/3442188.3445922

[4] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

[5] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. 2011. D^3 Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics* 17, 12 (2011), 2301–2309. https://doi.org/10.1109/TVCG.2011.185

[6] Robert-Mihai Botarleanu, Mihai Dascalu, Laura K. Allen, Scott Andrew Crossley, and Danielle S. McNamara. 2022. Multitask Summary Scoring with Longformers. In *Artificial Intelligence in Education*, Maria Mercedes Rodrigo, Noburu Matsuda, Alexandra I. Cristea, and Vania Dimitrova (Eds.). Springer International Publishing, Cham, 756–761. https://doi.org/10.1007/978-3-031-11644-5_79

[7] R.E. Boyatzis. 1998. *Transforming Qualitative Information: Thematic Analysis and Code Development.* SAGE Publications.

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 1877–1901. https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[9] Peter Brusilovsky, Sergey Sosnovsky, and Khushboo Thaker. 2022. The Return of Intelligent Textbooks. *AI Magazine* 43, 3 (Aug. 2022), 337–340. https://doi.org/10.1002/aaai.12061

[10] Yuanzhe Chen, Qing Chen, Mingqian Zhao, Sebastien Boyer, Kalyan Veeramachaneni, and Huamin Qu. 2016. DropoutSeer: Visualizing learning patterns in Massive Open Online Courses for dropout reasoning and prediction. In *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*. 111–120. https://doi.org/10.1109/VAST.2016.7883517

[11] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, Florence, Italy, 276–286. https://doi.org/10.18653/v1/W19-4828

[12] Scott A. Crossley, Minkyung Kim, Laura Allen, and Danielle McNamara. 2019. Automated Summarization Evaluation (ASE) Using Natural Language Processing Tools. In *Artificial Intelligence in Education*, Seiji Isotani, Eva Millán, Amy Ogan, Peter Hastings, Bruce McLaren, and Rose Luckin (Eds.). Springer International Publishing, Cham, 84–95. https://doi.org/10.1007/978-3-030-23204-7_8

[13] Joseph F. DeRose, Jiayao Wang, and Matthew Berger. 2021. Attention Flows: Analyzing and Comparing Attention Mechanisms in Language Models. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2021), 1160–1170. https://doi.org/10.1109/TVCG.2020.3028976

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[15] Niklas Elmqvist, Andrew Vande Moere, Hans-Christian Jetter, Daniel Cernea, Harald Reiterer, and TJ Jankun-Kelly. 2011. Fluid interaction for information visualization. *Information Visualization* 10, 4 (2011), 327–340. https://doi.org/10.1177/1473871611413180

[16] Scott R. Emmons, Robert P. Light, and Katy Börner. 2017. MOOC visual analytics: Empowering students, teachers, researchers, and platform developers of massively open online courses. *Journal of the Association for Information Science and Technology* 68, 10 (2017), 2350–2363. https://doi.org/10.1002/asi.23852

[17] David Galbraith and Veerle M. Baaijen. 2018. The Work of Writing: Raiding the Inarticulate. *Educational Psychologist* 53, 4 (2018), 238–257. https://doi.org/10.1080/00461520.2018.1505515

[18] Dilrukshi Gamage, Thomas Staubitz, and Mark Whiting. 2021. Peer assessment in MOOCs: Systematic literature review. *Distance Education* 42, 2 (2021), 268–289. https://doi.org/10.1080/01587919.2021.1911626

[19] Germain Garcia-Zanabria, Daniel A. Gutierrez-Pachas, Guillermo Camara-Chavez, Jorge Poco, and Erick Gomez-Nieto. 2022. SDA-Vis: A Visualization System for Student Dropout Analysis Based on Counterfactual Exploration. *Applied Sciences* 12, 12 (2022). https://doi.org/10.3390/app12125785

[20] Mor Geva, Avi Caciularu, Guy Dar, Paul Roit, Shoval Sadde, Micah Shlain, Bar Tamir, and Yoav Goldberg. 2022. LM-Debugger: An Interactive Tool for Inspection and Intervention in Transformer-Based Language Models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Abu Dhabi, UAE, 12–21. https://doi.org/10.18653/v1/2022.emnlp-demos.2

[21] Steve Graham and Karen R. Harris. 2015. Common Core State Standards and Writing: Introduction to the Special Issue. *The Elementary School Journal* 115, 4 (2015), 457–463. https://doi.org/10.1086/681963

[22] Steve Graham, Sharlene A. Kiuhara, and Meade MacKay. 2020. The Effects of Writing on Learning in Science, Social Studies, and Mathematics: A Meta-Analysis. *Review of Educational Research* 90, 2 (2020), 179–226. https://doi.org/10.3102/0034654320914744

[23] Martha H Head, John E Readence, and Ray R Buss. 1989. An examination of summary writing as a measure of reading comprehension. *Reading Research and Instruction* 28, 4 (1989), 1–11. https://doi.org/10.1080/19388078909557982

[24] Fred Hohman, Minsuk Kahng, Robert Pienta, and Duen Horng Chau. 2019. Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers. *IEEE Transactions on Visualization and Computer Graphics* 25, 8 (2019), 2674–2693. https://doi.org/10.1109/TVCG.2018.2843369

[25] Benjamin Hoover, Hendrik Strobelt, and Sebastian Gehrmann. 2020. exBERT: A Visual Analysis Tool to Explore Learned Representations in Transformer Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Association for Computational Linguistics, Online, 187–196. https://doi.org/10.18653/v1/2020.acl-demos.22

[26] Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 3543–3556. https://doi.org/10.18653/v1/N19-1357

[27] Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. 2023. ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and individual differences* 103 (2023), 102274.

[28] Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf. 2015. Principles of Explanatory Debugging to Personalize Interactive Machine Learning. In *Proceedings of the 20th International Conference on Intelligent User Interfaces* (Atlanta, Georgia, USA) *(IUI '15)*. Association for Computing Machinery, New York, NY, USA, 126–137. https://doi.org/10.1145/2678025.2701399

[29] Simone Leonardi, Diego Monti, Giuseppe Rizzo, and Maurizio Morisio. 2020. Multilingual transformer-based personality traits estimation. *Information* 11, 4 (2020), 179.

[30] Q Vera Liao and Jennifer Wortman Vaughan. 2023. AI Transparency in the Age of LLMs: A Human-Centered Research Roadmap. *arXiv preprint arXiv:2306.01941*

(2023).

[31] Shixia Liu, Xiting Wang, Mengchen Liu, and Jun Zhu. 2017. Towards better analysis of machine learning models: A visual analytics perspective. *Visual Informatics* 1, 1 (2017), 48–56. https://doi.org/10.1016/j.visinf.2017.01.006

[32] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[33] Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 4765–4774. http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[34] Wesley Morris, Scott Crossley, Langdon Holmes, Chaohua Ou, Danielle McNamara, and Mihai Dascalu. 2023. Using Large Language Models to Provide Formative Feedback in Intelligent Textbooks. In *Artificial Intelligence in Education. Posters and Late Breaking Results, Workshops and Tutorials, Industry and Innovation Tracks, Practitioners, Doctoral Consortium and Blue Sky*, Ning Wang, Genaro Rebolledo-Mendez, Vania Dimitrova, Noboru Matsuda, and Olga C. Santos (Eds.). Springer Nature Switzerland, Cham, 484–489. https://doi.org/10.1007/978-3-031-36336-8_75

[35] Ahmed A. Mubarak, Han Cao, Weizhen Zhang, and Wenli Zhang. 2021. Visual analytics of video-clickstream data and prediction of learners' performance using deep learning models in MOOCs' courses. *Computer Applications in Engineering Education* 29, 4 (2021), 710–732. https://doi.org/10.1002/cae.22328

[36] Nancy Nelson and James R. King. 2023. Discourse synthesis: Textual transformations in writing from sources. *Reading and Writing* 36, 4 (01 Apr 2023), 769–808. https://doi.org/10.1007/s11145-021-10243-5

[37] Christopher Michael Ormerod. 2022. Mapping between hidden states and features to validate automated essay scoring using DeBERTa models. *Psychological Test and Assessment Modeling* 64, 4 (2022), 495–526.

[38] Emily Phillips Galloway and Paola Uccelli. 2019. Beyond reading comprehension: exploring the additional contribution of Core Academic Language Skills to early adolescents' written summaries. *Reading and Writing* 32, 3 (01 Mar 2019), 729–759. https://doi.org/10.1007/s11145-018-9880-3

[39] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21, 140 (2020), 1–67.

[40] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 1135–1144. https://doi.org/10.1145/2939672.2939778

[41] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 1135–1144. https://doi.org/10.1145/2939672.2939778

[42] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A Primer in BERTology: What We Know About How BERT Works. *Transactions of the Association for Computational Linguistics* 8 (2020), 842–866. https://doi.org/10.1162/tacl_a_00349

[43] Michael Sedlmair, Miriah Meyer, and Tamara Munzner. 2012. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2431–2440. https://doi.org/10.1109/TVCG.2012.213

[44] Yeonju Jang Seongyune Choi and Hyeoncheol Kim. 2023. Influence of Pedagogical Beliefs and Perceived Trust on Teachers' Acceptance of Educational Artificial Intelligence Tools. *International Journal of Human–Computer Interaction* 39, 4 (2023), 910–922. https://doi.org/10.1080/10447318.2022.2049145 arXiv:https://doi.org/10.1080/10447318.2022.2049145

[45] Angélica M. Silva and Roberto Limongi. 2019. Writing to learn increases long-term memory consolidation: A mental-chronometry and computational-modeling study of "Epistemic writing". *Journal of Writing Research* 11, 1 (Jun. 2019), 211–243. https://doi.org/10.17239/jowr-2019.11.01.07

[46] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615* (2022).

[47] Kenneth Steimel and Brian Riordan. 2020. Towards instance-based content scoring with pre-trained transformer models. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, Vol. 34.

[48] Hendrik Strobelt, Sebastian Gehrmann, Michael Behrisch, Adam Perer, Hanspeter Pfister, and Alexander M. Rush. 2019. Seq2seq-Vis: A Visual Debugging Tool for Sequence-to-Sequence Models. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 353–363. https://doi.org/10.1109/TVCG.2018.2865044

[49] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 3319–3328. https://proceedings.mlr.press/v70/sundararajan17a.html

[50] Ian Tenney, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, Ellen Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif, and Ann Yuan. 2020. The Language Interpretability Tool: Extensible, Interactive Visualizations and Analysis for NLP Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online, 107–118. https://doi.org/10.18653/v1/2020.emnlp-demos.15

[51] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[53] Camilo Vieira, Paul Parsons, and Vetria Byrd. 2018. Visual learning analytics of educational data: A systematic literature review and research agenda. *Computers & Education* 122 (2018), 119–135. https://doi.org/10.1016/j.compedu.2018.03.018

[54] Jesse Vig. 2019. A Multiscale Visualization of Attention in the Transformer Model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Association for Computational Linguistics, Florence, Italy, 37–42. https://doi.org/10.18653/v1/P19-3007

[55] Eric Wallace, Pedro Rodriguez, Shi Feng, Ikuya Yamada, and Jordan Boyd-Graber. 2019. Trick Me If You Can: Human-in-the-Loop Generation of Adversarial Examples for Question Answering. *Transactions of the Association for Computational Linguistics* 7 (2019), 387–401. https://doi.org/10.1162/tacl_a_00279

[56] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, Brussels, Belgium, 353–355. https://doi.org/10.18653/v1/W18-5446

[57] Changhan Wang, Anirudh Jain, Danlu Chen, and Jiatao Gu. 2019. VizSeq: a visual analysis toolkit for text generation tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*. Association for Computational Linguistics, Hong Kong, China, 253–258. https://doi.org/10.18653/v1/D19-3043

[58] Zijie J. Wang, Dongjin Choi, Shenyu Xu, and Diyi Yang. 2021. Putting Humans in the Natural Language Processing Loop: A Survey. In *Proceedings of the First Workshop on Bridging Human–Computer Interaction and Natural Language Processing*. Association for Computational Linguistics, Online, 47–52. https://aclanthology.org/2021.hcinlp-1.8

[59] Zijie J. Wang, Robert Turko, and Duen Horng Chau. 2021. Dodrio: Exploring Transformer Models with Interactive Visualization. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online, 132–141. https://zijie.wang/papers/dodrio/

[60] Zijie J. Wang, Robert Turko, Omar Shaikh, Haekyu Park, Nilaksh Das, Fred Hohman, Minsuk Kahng, and Duen Horng Polo Chau. 2021. CNN Explainer: Learning Convolutional Neural Networks with Interactive Visualization. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2021), 1396–1406. https://doi.org/10.1109/TVCG.2020.3030418

[61] Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not Explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 11–20. https://doi.org/10.18653/v1/D19-1002

[62] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online, 38–45. https://doi.org/10.18653/v1/2020.emnlp-demos.6

[63] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453* (2023).

[64] Gefei Zhang, Zihao Zhu, Sujia Zhu, Ronghua Liang, and Guodao Sun. 2022. Towards a better understanding of the role of visualization in online learning: A review. *Visual Informatics* 6, 4 (2022), 22–33. https://doi.org/10.1016/j.visinf.2022.09.002

[65] Huijie Zhang, Jialu Dong, Cheng Lv, Yiming Lin, and Jinghan Bai. 2023. Visual analytics of potential dropout behavior patterns in online learning based on counterfactual explanation. *Journal of Visualization* 26, 3 (01 Jun 2023), 723–741. https://doi.org/10.1007/s12650-022-00899-8