Research Methods in Applied Linguistics 3 (2024) 100153

Contents lists available at ScienceDirect

Research Methods in Applied Linguistics

journal homepage: www.elsevier.com/locate/rmal

Building custom NLP tools to annotate discourse-functional features for second language writing research: A tutorial

Masaki Eguchi [a,*], Kristopher Kyle [b]

[a] Waseda University, Japan
[b] University of Oregon, USA

ARTICLE INFO

Keywords:
Natural language processing
Appraisal analysis
Epistemic stance
L2 writing
Corpus annotation

ABSTRACT

The current tutorial paper describes a process of developing a custom natural language processing model with a particular focus on a discourse annotation task. After an overview of recent developments in natural language processing (NLP), the paper discusses the development of the Engagement Analyzer (Eguchi & Kyle, 2023), focusing on corpus annotation, the machine learning model, model training, evaluation, and dissemination. A step-by-step tutorial of this process via the spaCy Python package is provided. The paper highlights the feasibility of developing custom NLP tools to enhance the scalability and replicability of the annotation of context-sensitive linguistic features in L2 writing research.

Introduction

Second Language Acquisition (SLA) researchers are often interested in describing and understanding second language production in terms of features that distinguish L2 proficiency levels and how performance changes as a result of increased exposure and prolonged learning (Ortega, 2009). To make an evidence-based claim about these facets of L2 learning and development, researchers need a systematic means to gauge learner performance (Norris & Ortega, 2003). As such, the measurement of written/spoken L2 performances has been a central topic in second language (L2) research (Biber et al., 2020; Ellis & Barkhuizen, 2005; Housen et al., 2012; Kuiken & Vedder, 2017; Norris & Ortega, 2003, 2009; Wolfe-Quintero et al., 1998). If reliable and valid, such measurement frameworks not only enrich the theoretical claims by enhancing research comparability but also inform instructional and assessment practices by offering toolkits for practitioners to facilitate their decision-making.

Perhaps one of the most widely recognized and used frameworks in SLA research is the complexity, accuracy, (lexis,) and fluency (CAF or CALF) framework (Housen et al., 2012; Skehan, 2009; Wolfe-Quintero et al., 1998). CAF measures are often used to describe the changes in learner performances due to task manipulation and time spent studying a language. They are often interpreted through cognitive approaches to task-based performance (e.g., Kormos, 2011). Recently, however, researchers have become more aware of the limitations of CAF measures (e.g., Pallotti, 2009), and consequently, more research has focused on functional dimensions of language performance, conceptualized as functional or communicative adequacy (Kuiken & Vedder, 2017). With such a trend, objective measures that focus on the functional dimension of writing are also proposed and used to describe the characteristics of learner language use (Bax et al., 2019; Qin & Liu, 2024; Yoon, 2017).

Natural Language Processing (NLP) technologies have been integral to the development of automatic analysis of the features of

* Corresponding author.
E-mail addresses: meguchi@aoni.waseda.jp (M. Eguchi), kkyle2@uoregon.edu (K. Kyle).

https://doi.org/10.1016/j.rmal.2024.100153
Received 1 June 2024; Received in revised form 6 September 2024; Accepted 6 September 2024
Available online 18 September 2024

language use. Tools such as Coh-metrix (Graesser et al., 2004), the Lexical Complexity Analyzer (Lu, 2012), and the Tool for the Automatic Analysis of Lexical Sophistication (TAALES) (Kyle et al., 2018) have contributed to the execution of empirical studies on lexical and syntactic complexity and other textual features at scale. Many corpus tools used to analyze (L2) written production adapt classic NLP tasks such as part-of-speech tagging (Biber, 2004), constituency parsing (Park & Lu, 2015), dependency parsing (Kyle, 2016; Kyle & Eguchi, 2021; Paquot, 2019), and word embeddings (Crossley et al., 2019). These tools have not only significantly helped reduce the labor-intensive nature of linguistic coding for individual studies but also enhanced research replicability and reproducibility across different research contexts.

While more and more automated tools are accessible to researchers, manual annotation can still be a severe obstacle for researchers whose work concerns less investigated and more context-dependent discourse features. Over-reliance on already-existing automated tools may also lead to imbalances in research that favor more readily accessible surface-level linguistic features. Methodological innovation is needed to address such potential imbalances by developing automatic annotation systems that can identify contextually dependent features that are of theoretical importance in L2 writing research. While this automation process has been largely constrained by the limitations of corpus-linguistic and NLP tools, recent breakthroughs in NLP methods to model and understand natural language have lowered barriers to creating automated tools to undertake context-aware annotation of linguistic features.

The overarching goal of this tutorial is to showcase the process of developing a custom NLP tool specifically designed to automatically identify and annotate discourse functions of rhetorical stance-taking features, which has been demonstrated as one of the essential aspects of L2 writing (Biber, 2006; Chang & Schleppegrell, 2011; Crosthwaite & Jiang, 2017). Taking Eguchi and Kyle (2023) as a working example, we illustrate the entire process of designing a new NLP task, annotating corpus data, training and evaluating machine-learning models, and disseminating the tool for research and education. Our goal is to demonstrate that a probabilistic machine-learning approach would be a viable option for overcoming challenges and obstacles researchers may face in their research when annotating functionally ambiguous (poly-functional) linguistic features in discourse.

We hope that the current tutorial serves as a useful conceptual introduction to the machine-learning approach to develop a new NLP tool. Thus, the intended readers of this tutorial include students and researchers in applied linguistics with a range of experiences in corpus linguistics and NLP. As an introductory tutorial, knowledge of programming languages is not required to understand the conceptual issues covered in the article; however, basic knowledge of Python programming language is required to follow the online tutorial to reproduce the machine-learning experiments. Suggested readings are provided at the end of this article for those who would like to gain more conceptual knowledge and practical skills to apply NLP approaches to their research.

Preliminaries

Common natural language processing tasks

Over the years, NLP researchers have defined various tasks depending on the goals of the linguistic analysis being conducted. Typical NLP tasks include but are not limited to part-of-speech (POS) tagging, dependency parsing, text classification, and named entity recognition. In NLP, a task can be described in terms of characteristics of input, output, unit of analysis, and the type of information being identified, extracted, or categorized. Readers are referred to Jurafsky and Martin (2009)) for an in-depth treatment of typical NLP tasks and the standard algorithms used to implement them.

Part-of-speech (POS) tagging is a task wherein a part-of-speech label is assigned to each token in a set of sentences. At a more general level, this is a *sequence labeling* task, where each word in the input sequence is tagged for pre-determined categories. A range of L2 studies relies on POS taggers to describe language proficiency across registers as a part of the multidimensional analysis (Biber & Gray, 2013) and disambiguate the grammatical functions of bigrams, such as ADJ + NOUN (Bestgen & Granger, 2018).

Dependency parsing is the task of finding binary pairs of syntactically related tokens (i.e., lexical nodes) in given sentences and categorizing their syntactic relationships (e.g., nominal subject, clausal subject, direct object). Dependency parsing has gained popularity in second-language research through the introduction of EF-CAMDAT (Shatz, 2020), which offers a dependency annotation in both cross-sectional and longitudinal second-language writing corpus (Huang et al., 2018). Dependency parsing has also been applied in extracting lexical collocations, which may be dislocated due to complex syntactic structures (e.g., Paquot, 2019). Measures of syntactic sophistication also use dependency parsing to extract the patterns of syntactic arguments (Kyle, 2016). For those who are interested in understanding and applying POS taggers or dependency parsing specifically, see recent comparative analyses across both off-the-shelf and custom-trained models (Kyle & Eguchi, 2024).

Named Entity Recognition (NER) is a task of identifying a named entity (i.e., London, US dollar, etc.) and assigning their entity categories (i.e., London → geopolitical place, US dollar → currency, etc.). This is another sequence labeling task, where the algorithm first finds the correct chunking boundaries (i.e., start and end tokens) for the entity and then categorizes them into preset categories. An NER-like sequence labeling task has been used in Kyle and Sung (2023) to identify and disambiguate nine argument-structure-constructions based on construction grammar (A. E. Goldberg, 2003), including caused-motion (e.g., *[The body]$_{agent}$ brings [stability]$_{theme}$ [to the region]$_{goal}$*) and ditransitive (e.g., *[You]$_{agent}$ feed [your rabbits]$_{recipient}$ [non-veg items]$_{theme}$*) constructions.

Span categorization is another type of sequence labeling task (Gu et al., 2022; Papay et al., 2020). Like NER, span categorization can detect contiguous sequences of expression with any predetermined categories (e.g., specific entities, events, or linguistic construction). What distinguishes this task from NER is that span categorization can identify nested or overlapping spans, hence offering a more flexible option. In second language research, Eguchi and Kyle (2023) applied this NLP task to identify expressions related to epistemic stances, which is the working example for the current tutorial.

Supervised machine learning for natural language processing

To date, the classic NLP tasks described above usually rely on *supervised machine-learning* approaches as a framework (but see S. Wang et al., 2023). Supervised machine learning is a paradigm in machine learning where the researcher provides a computer algorithm with many examples of "correct" input-output pairs and has it learn (non-)linear functions to minimize errors in prediction. In this approach, instead of hard coding *deterministic* rules (such as if-then statements), the researchers have the algorithm learn *probabilistic* rules out of the provided input-output pairs. The outcome—a machine-learning model—can be thought of as a non-linear "regression" algorithm that takes an input and predicts its output label.

Usually, a single "training" process is performed iteratively through multiple rounds, where a subset of examples from the training data is used to make tentative predictions using the algorithm at time *t*, and the amount of deviation between this prediction and the "correct" answers is calculated as a *loss*. This information about loss is then used to update the algorithm's parameter weights (a process called *backpropagation*) so that the updated algorithm at time $t + 1$ will perform slightly better on the next batch of the training examples. This process is repeated incrementally to minimize the loss gradually, known as *stochastic gradient descent*, which is analogous to the process of finding a global solution in maximum likelihood estimation in statistics.

When researchers train a model, they can adjust some controllable parameters of the model, such as the learning rate (the rate at which the model parameter is updated) and batch size (the number of instances seen by the algorithm at once to update the parameters). These *hyperparameters* are a part of the model and can influence its performance. The list of hyperparameters depends on the type of model, and the detailed discussion of it goes beyond the scope of this section, although the term is briefly mentioned in the tutorial.

To date, widely used machine-learning algorithms (e.g., multi-layer perceptron, deep neural network) can learn highly complex functions that minimize the loss and produce highly accurate predictions on training data relatively easily. However, one downside of modern ML algorithms includes learning a too-complex function that performs almost perfectly on the training examples but does not generalize the prediction to unseen data (i.e., *overfitting*). To prevent overfitting, ML researchers usually split the entire dataset into three parts (i.e., *train, validation* also referred to as *development*, and *test*). This three-way split data is used in the following manner. During the training iteration, the performance of the model is checked occasionally on the development set (e.g., after 100–200 parameter updates). At the end of the training, the best-performing model parameter weights are retained so that the prediction accuracy (and the loss) on the development set is maximized. Once a single model is trained, its performance can be tested on the test set (held-out data with completely new instances of the target task). For introductory text on supervised machine-learning methods in NLP, see Goldberg (2017).

Evaluation metrics of NLP models

When training and evaluating machine-learning models, it is essential to determine appropriate evaluation metrics. In classic NLP tasks involving classification (see above), metrics evaluating the correct identification and categorization are often used—e.g., Precision, Recall, F1, and Cohen's Kappa.

Precision, recall, and F1 metrics

Precision and Recall are calculated for each category using the following formula:

$$Precision = \frac{True\ Positives}{(True\ Positives + False\ Positives)}$$

$$Recall = \frac{True\ Positives}{(True\ Positives + False\ Negatives)}$$

where True Positives (TP) are defined as the number of data points correctly classified for a given category; False Positives (FP) concern the number of data points incorrectly classified as a given category, and False Negatives (FN) count the number of candidates incorrectly labeled as other categories. Generally, the fewer FPs identified by the system, the greater will be the precision. Similarly, the fewer the true items missed by the system, the higher the recall score will be. F1 score is a harmonic mean of recall and precision calculated using the following formula:

$$F1 = 2\frac{Precision\ \cdot Recall}{Precision + Recall}$$

Thus, the F1 score is considered to be a balanced single metric to summarize Precision and Recall scores and is often introduced as a metric to evaluate overall system performance in the machine learning literature (see Dror et al., 2020; Thampi, 2022).

Cohen's kappa

Although precision, recall, and F1 scores are widely used to evaluate a model's performance, they do not consider the chance-level agreement between gold tags and system predictions. This information can be obtained with Cohen's kappa coefficient (Cohen, 1960). As explained by McHugh (2012) and Landis and Koch (1977), Cohen's kappa coefficient (Cohen, 1960) ranges from −1 to +1. It adjusts observed agreement rates based on expected agreement rates for a given contingency table, as in the following formula:

$$Kappa = \frac{\Pr(actual) - \Pr(expected)}{1 - \Pr(expected)}$$

Where the expected agreement rate is calculated using the total counts of independent observations and marginal sums of the contingency table. Taking a simple 2-by-2 contingency table as a hypothetical example, the expected agreement is calculated as (after McHugh, 2012):

$$\Pr(expected) = \frac{\left(\frac{Column\ 1\ marginal * Row\ 1\ marginal}{n}\right) + \left(\frac{Column2\ margninal * Row2\ marginal}{n}\right)}{n}$$

Suppose that the hypothetical chance level agreement is 0.20 and the observed agreement rate is 0.74; then the kappa coefficient would be:

$$kappa = \frac{.74 - .20}{1 - .20} = .675$$

Resulting in an adjusted rate of agreement given the chance level agreement.

Table 1 summarizes two existing benchmarks to interpret the level of agreement based on Cohen's kappa values. In Landis and Koch's (1977) classic benchmark, a kappa coefficient of above 0.60 is considered substantial. In contrast, McHugh (2012) proposes a more conservative benchmark, treating the 0.60–0.79 range as moderate agreement and 0.80–0.89 as strong agreement. Following these two guidelines, we consider a kappa value below 0.6 to be "inadequate," a kappa between 0.60 and 0.80 is "moderate-to-substantial," while a value over 0.80 indicates "strong" reliability of an instrument.

K-fold cross-validation

In machine-learning experiments, the use of a single three-way data split may not yield a generalizable result, particularly when the entire dataset is small and only one version of test data may not represent the domain well. In such cases, *K*-fold cross-validation is often conducted to see whether and to what extent the evaluation metrics (e.g., Precision, Recall, and F1 scores) may depend on particular choices of splits (Y. Goldberg, 2017). In K-fold cross-validation, the entire dataset is usually split into *K*-fold data, as shown in Fig. 1 (illustrating a 5-fold CV), and a single model is trained *K* times with alternating sets of training, development, and test data. This results in *K* sets of scores—five sets of scores for each of the development and test data. Subsequently, the researchers can examine the central tendency and the variability across the folds to evaluate the robustness of the model. *K*-fold CV provides realistic estimates of the model's accuracy by allowing one to utilize the available data efficiently.

The breakthroughs in NLP—The transformer architecture, self-supervised pre-training objective, and transfer learning

The rise of generative models such as Generative Pretrained Transformers (GPT; Brown et al., 2020; Radford et al., 2018) has made NLP methods more and more accessible to laypersons. An emerging line of research demonstrates the capability of such models in linguistic annotation (e.g., Kim & Lu, 2024; S. Wang et al., 2023). While the current tutorial does not cover generative language models, the underlying technological breakthroughs are shared in the approach taken in this tutorial to annotate discourse features automatically. This breakthrough can be explained by at least the following four factors—the Transformer architecture, self-supervised pre-training, the amount of pre-training data, and transfer learning. Automatic annotation of discourse features, which requires a greater understanding of semantic (or even pragmatic) features of the given input sentences, has become increasingly realistic thanks to these technologies.

In essence, the Transformer (Vaswani et al., 2017) is a class of deep-learning architecture that enables NLP models to *attend* to "useful" co-textual information when processing textual inputs. This attention mechanism (or self-attention, more precisely) is often explained using examples from semantic disambiguation tasks. Consider the following two uses of the word *charge*—(a) *The host will **charge** you for any damages to the property*, and (b) *Feel free to **charge** your phone before you leave*. As with the famous quote, "[y]ou shall know a word by the company it keeps" (Firth, 1957, p. 11), disambiguating the two senses requires co-textual information. For example, the colligations/collocations "*charge for any damage*" in the first example and "*charge your phone*" in the second example allow the speakers to narrow down the interpretations of the word *charge* in the respective contexts. The self-attention mechanism of the Transformer architecture attempts to approximate this process—it considers a *weighted sum* of the surrounding words to understand the

Table 1
Two existing benchmarks for interpreting Cohen's Kappa coefficient.

Kappa values			Landis and Koch's (1977) benchmark	McHugh's (2012) revised benchmark
< 0.00			Poor	None
.00	–	.20	Slight	None
.21	–	.39	Fair	Minimal
.40	–	.59	Moderate	Weak
.60	–	.79	Substantial	Moderate
.80	–	.89	Almost Perfect	Strong
> 0.9			Almost Perfect	Almost Perfect

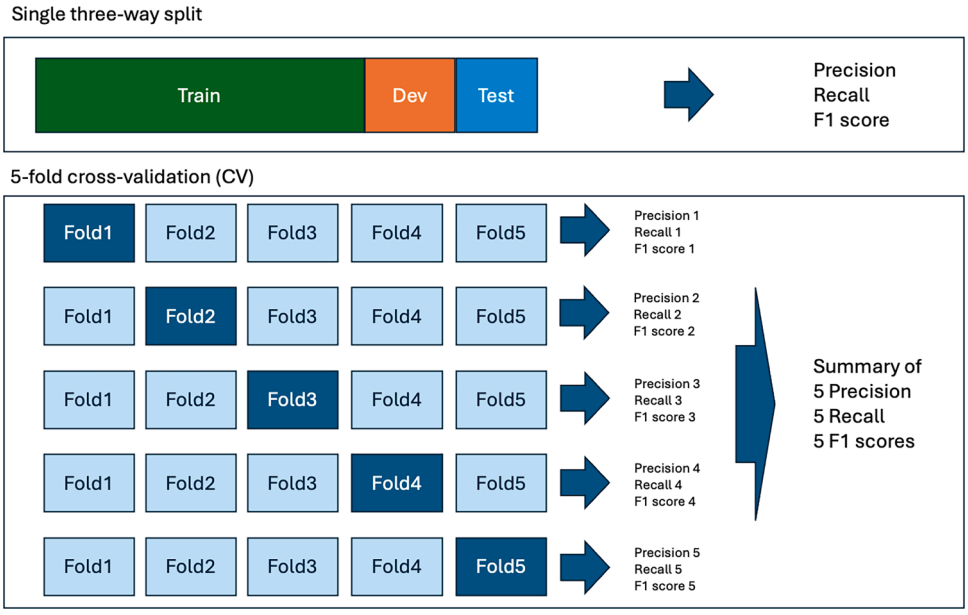

Fig. 1. Illustration of three-way split and 5-fold cross-validation.

meaning of the target word in context.

How does a Transformer model learn where to attend? This knowledge is obtained when they are trained on a massive number of texts without any human labels (called *self-supervised pre-training*). This pre-training objective does not require correct labels from humans but rather massive corpora. In principle, generative models such as GPT are trained to predict the next word(s) (typically one word at a time) given the preceding sequences (Radford et al., 2018). Another commonly used pre-training task includes masked language modeling (Devlin et al., 2019). Intuitively, the masked language objective can be thought of as solving a massive number of cloze (or gap-filling) tasks throughout the training corpus. While iteratively reconstructing original texts (as in the stochastic gradient descent described earlier), Transformer models can learn latent "black-box" representations (or attention weights) of various linguistic features (e.g., word senses, morphology, syntax, semantic roles, discourse features) without any explicit gold-standard input from humans (Clark et al., 2019). Once they are pre-trained, the masked language models can convert a sequence of input text into contextually rich vector representation (i.e., a series of numbers representing word meanings in context).

A contextual vector obtained through a pre-trained Transformer model is then used as the input for subsequent machine-learning algorithms that perform desired NLP tasks. This strategy to adapt pre-trained models to downstream tasks is called *transfer learning* or *fine-tuning*. The current demonstration deals with an approach to fine-tune a family of pre-trained masked language models, such as the **B**idirectional **E**ncoder **R**epresentations from **T**ransformers (Devlin et al., 2019) and the **R**obustly **o**ptimized **BERT** pretraining **a**pproach (RoBERTa; Liu et al., 2019), to perform a discourse annotation task. For an accessible introduction to the Transformer architecture, see Hagiwara (2022).

This tutorial

In this tutorial article, we provide a step-by-step tutorial on how to develop a custom NLP model to automatically annotate rhetorical stance expressions in academic English, using Eguchi and Kyle's (2023) Engagement Analyzer as the working example (for demo, see https://huggingface.co/spaces/egumasa/engagement-analyzer-demo). The Engagement Analyzer is a span identification model that takes an English text as input and identifies expressions that enact the author's epistemic stance(s) on a topic of discussion and functional categories of those, following the Engagement framework by Martin and White (2005) (see below). As will be illustrated, this discourse-analytic approach to the analysis poses challenges in identifying the discourse functions of a given expression in context using traditional corpus-linguistic tools such as KWIC search or regular expressions. In what follows, we first introduce the working example by providing the overview of the discourse analytic framework of Engagement, followed by the overall NLP architecture of the tool. We then proceed with a step-by-step tutorial on how to develop a custom NLP tool. We also provide detailed online tutorials for those who are interested in reproducing the machine learning experiment described in this article through Google Collaboratory (https://github.com/egumasa/engagement-analyzer-train). The annotation schemes developed out of this project can be accessed at the following website (https://egumasa.github.io/engagement-annotation-project/).

The framework of engagement—The discourse framework

Essentially, engagement is a discourse-analytic framework that allows researchers to examine how an author of a text positions

themselves with respect to potential alternative viewpoints (see details below). Martin and White (2005) explain how a speaker or writer uses three broad classes of discourse meaning—*attitude, engagement*, and *graduation*—to express interpersonal meaning. Attitude concerns the expressions of affective stances, such as emotions, feelings, judgment, and evaluations. Graduation is related to intensification and grading of attitude and engagement (e.g., *a bit* < *somewhat* < *very* < *completely; I suspect* < *I believe* < *I am convinced*). Finally, engagement, the focus of this paper, is concerned with epistemic stance, or more precisely, "locutions which provide the means for the authorial voice to position itself with respect to, and hence to 'engage' with, the other voices and alternative positions construed as being in play in the current communicative context" (Martin & White, 2005, p. 94).

Fig. 2 describes the taxonomic tree of the engagement system. Starting from the top-level branch, we can distinguish between *monogloss* and *heterogloss*. This binary decision concerns whether the immediate utterance recognizes alternative positions. In monogloss, the writer does not recognize any alternative views and presents the idea/ event as if it is a fact (e.g., "The banks have been greedy"; Martin & White, 2005). On the other hand, a heteroglossic utterance includes various ways in which writers display their recognition of possible alternatives to what is being discussed (e.g., "I *speculate* that the banks have been greedy," "I *read somewhere* that the banks have been greedy", "It is *unlikely* that the banks have been greedy", etc.).

A heteroglossic statement can be distinguished in terms of whether the writer makes some allowances for alternative standpoints in the immediate discourse (*expansion*) or does not allow such room for negotiation (*contraction*). Expansion strategies include discourse strategies that (a) increase the tentativeness of the statement (*Entertain*) and (b) attribute the idea to external sources (*Attribute*). With Entertain, a writer can use lexico-grammatical items such as modal verbs (*can, may*) and mental verbs (I *believe*) to entertain other possible alternatives. In Attribute, writers mention what is reported in (un)identified external sources (e.g., the paper mentioned, it is believed that).

In contrast to expansion strategies, contraction includes discourse strategies where writers attempt to close the dialogic space for discussion. This is done by either rejecting other viewpoints *(disclaim*) or bolstering their own views (*proclaim*). In disclaim, writers can *Deny* the reliability of a particular point of view (e.g., That is NOT correct) or *Counter* the alternative ideas (e.g., Although the paper may be right, there is another possibility). In proclaim, writers attempt to enhance the validity of their views by (a) formulating the locution assuming that it will be easily accepted by putative readers (*Concur*; e.g., as you know, of course, surely), (b) showing extra commitment to the validity of their views (*Pronounce*; we conclude that), or (c) presenting other's perspective/ data/ claims as correct and reliable and aligning to them (*Endorse*).

Although not included in Martin and White's (2005) system, *Justify* is another strategy considered in White's (2003) system. According to White (2003), it concerns "formulations which construe a particular type of consequentiality, namely those by which non-'factual' propositions (for example, attitudinal evaluations, directives/ recommendation, predictions and so on) are justified, substantiated or otherwise argued for" (p. 274). Typical resources include conjunctions such as *therefore, thus, accordingly, because*, and *for this reason* (White, 2003, p. 274). Table 2 summarizes the ten engagement strategies considered for annotation.

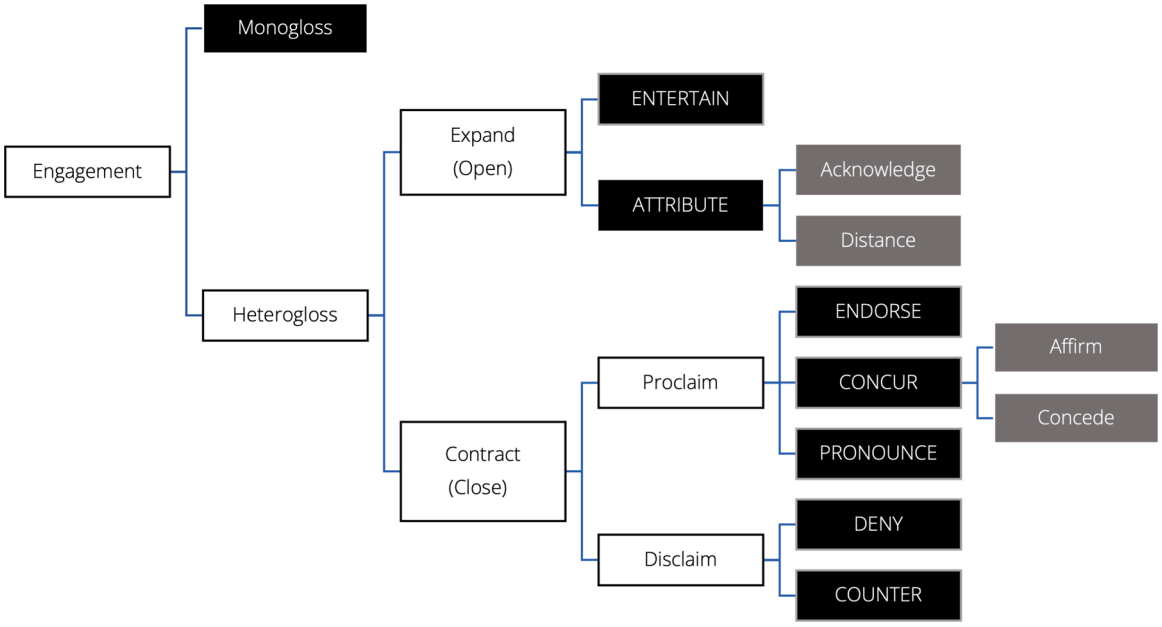

Fig. 2. The engagement system (Adapted from Martin & White, 2005).
Note. Black boxes show categories used for annotation of the corpus; Gray boxes indicate categories mentioned in Martin and White (2005).

Table 2
Engagement category layer tags (Adapted from Wu, 2007; Xu, 2020).

Macro strategy	Engagement strategy	Description	Examples of prototypical lexico-grammatical realizations (see Chang & Schleppegrell, 2011)
Contraction	Disclaim: Deny	An utterance which invokes a contrary position but which at the same time rejects it directly. The contrary position is hence given very little dialogic space.	• Negative particles (e.g., *not, never*)
Contraction	Disclaim: Counter	An utterance which expresses the present proposition as replacing and thus 'countering' another proposition which would have been expected.	• Conjunctions (e.g., *but*) • Adverbials (e.g., *however*) • Adverbial clauses (e.g., *although*)
Contraction	Proclaim: Concur	An utterance which shows a writer's expectation/ assumption that putative readers will agree with the proposition and/or have the same knowledge.	• Adverbials (e.g., *indeed, of course*) • Adverbial clauses (e.g., *as one expects*) • Display questions; tag questions
Contraction	Proclaim: Pronounce	An utterance which expresses a strong level of writer commitment through the author's explicit emphasis and interpolation, thereby closing down the dialogic space.	• Mental/Communication verbs (e.g., I *contend*, we *conclude*, I *propose*) • Emphatic do (e.g., I *do* believe that) • Modal attributes (e.g., *it is evident* that)
Contraction	Proclaim: Endorse	An utterance which refers to external sources as warrantable, undeniable, and/or reliable. It expresses the writer's alignment with and endorsement of an attributed proposition. As such, the dialogic space is somewhat narrowed.	• Reporting verb (e.g., Kyle (2020) *demonstrated* that)
Expansion	Entertain	An utterance which indicates the author's position but as only one possibility amongst others, thereby opening up dialogic space.	• Modal verbs (mainly epistemic and deontic modals; e.g., may, would; Palmer, 2001) • Mental/Communication verbs (e.g., I *think/ suppose*, we *suggest*) • Adverbials (e.g., *perhaps, probably*) • Adverbial clauses (e.g., *unless, when, if*) • Modal attributes (e.g., *it is likely that*) • Evidentials (e.g., *seem, apparently*)
Expansion	Attribute	An utterance which signifies dialogic space as the writer attributes the proposition to an external source.	• Adverbials (e.g., *reportedly*) • Reporting verbs (e.g., *they argue/ believe*)
Monogloss	Monogloss	An utterance which does not employ any value of engagement. Such an utterance ignores the dialogic potential in an utterance.	• Present-tense verbs • Lacks of any other engagement strategies
Auxiliary	Justify	An utterance which engages in persuasion through justification or substantiation.	• Adverbials (e.g., *therefore, for this reason*) • Conjunctions (e.g., *because*)

Note. Adapted from Martin and White (2005), Wu (2007), and Xu (2020).

Engagement analyzer—The proposed NLP approach

The engagement analyzer is a span categorization model trained with the Spancat component of spaCy (Honnibal et al., 2020) in Python. It takes raw English text as input and produces a list of sequences of tokens (i.e., spans) with a predicted engagement label for each. Table 3 shows the expected output given the following input sequence—*My guess is that you would probably not believe in this approach yet*. The main outputs of the model are a) the span of Engagement resources (i.e., *My guess is, would probably*, and *not believe*) and b) a predicted Engagement function class (i.e., ENTERTAIN, ENTERTAIN, and DENY, respectively). The model also produces the Start and End tokens and ID for unique identifiers.

Fig. 3 illustrates the series of operations the Engagement Analyzer applies to the input to arrive at the outcome prediction. It consists of three operations: token embedder, span candidate suggester, and span categorizer. Each component is briefly explained using the example below.

When the model receives an input (e.g., *My guess is that you would probably not believe in this approach yet.*), the token embedder converts the raw textual representation (i.e., surface-form) to a 768-dimensional vector representing the "underlying meaning" of each token in the input text. In Engagement Analyzer, this embedding operation is performed with a pre-trained Transformer-based language model called RoBERTa (Liu et al., 2019), which produces vector representations of each token by taking the surrounding co-textual information into account—a weighted sum of representations in all tokens in the input sequence. It is the use of a Transformer-based pre-trained model for token embedder that produces a contextually aware latent representation of meaning that is useful for the disambiguation of rhetorical features in the Engagement Analyzer.

In the second step, the span candidate suggester slices the input text into a series of textual spans using two algorithms (i.e., n-gram, and dependency subtree). The n-gram suggester produces a series of n-grams as a candidate span for classification; the dependency

Table 3
An example output of engagement analyzer.

ID	Text	Predicted Label	Start token	End token
0	My guess is	ENTERTAIN	0	3
1	would probably	ENTERTAIN	5	7
2	not believe	DENY	7	9

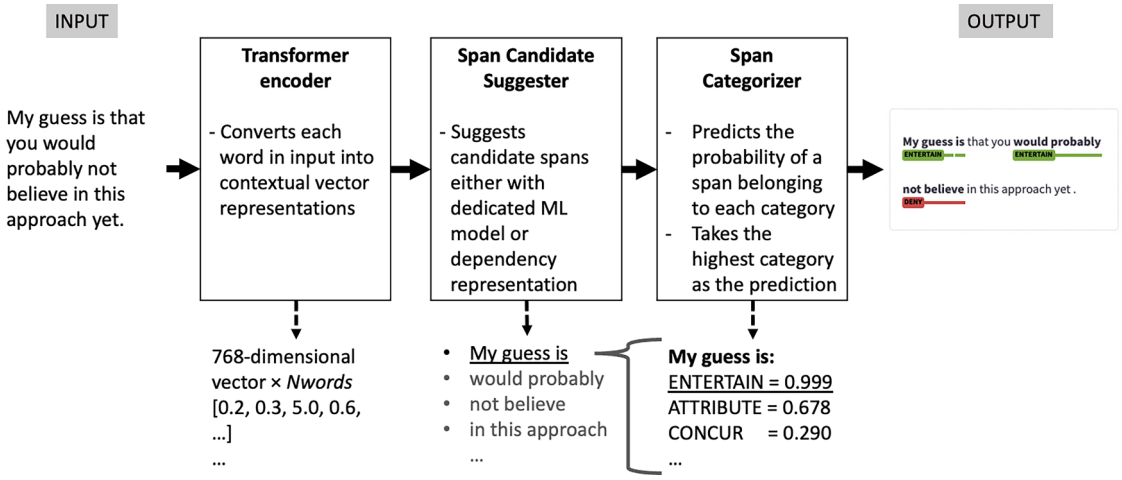

Fig. 3. The NLP Pipeline component of the Engagement Analyzer.

subtree suggester produces a series of grammatical constituencies (adverbial clauses, prepositional phrases) as candidates for classification. The goal of this step is to search for possible textual spans as greedily as possible so as to reduce the false negative rate in the final model. Table 4 provides an illustration of subtree suggester output.

The final component, span categorizer, takes the transformer embeddings and the candidate span information from the previous two steps and makes a final prediction as to whether a candidate span falls into the characteristics of Engagement resource and, if so, which of the Engagement categories it enacts. Details of these operations are not explained for simplicity, but the span categorizer essentially (a) summarizes the transformer embeddings for multi-word span by performing some pooling operations (e.g., taking the first and last words, or a mean of all tokens in a span), (b) apply non-linear activation functions to learn complex relationships between the embedding and output label, and (c) apply logistic regression to predict the probability of each class.

This model architecture allows predictions of functional categories of Engagement while considering their meanings in a given co-textual information. For example, to understand the rhetorical function of "*suggest*" in a given context—whether it enacts ATTRIBUTE or ENTERTAIN—our expert knowledge suggests that we should look for the Agent/Experiencer of this sentence (at the level of the semantic role). The Engagement Analyzer attempts to learn these kind of probabilistic rules (i.e., how to attend to the co-textual information) through optimizing the prediction of input-output pairs during the training. Fig. 4 provides an illustrative predictive output of the Engagement Analyzer, analyzing a published excerpt by Chang and Schleppegrell (2011). In the proposed probabilistic ML approach, patterns of engagement are learned through the input and output correspondences across the training examples.

How to build a custom NLP tool—The step-by-step tutorial

With the theoretical and methodological frameworks for the working example explained, the current section proceeds with the step-by-step tutorial on how to design, develop, and evaluate a custom NLP tool. Table 5 outlines the steps covered in the current tutorial. Each step is headed with general purposes, goals, and useful guiding questions. It is important to note that the presented steps are intended as a general guideline to help streamline the research planning; however, the actual research process likely involves iterative processes going back and forth to ensure that the desired result can be obtained.

The first half of the current procedure (particularly Steps 2–5) is closely modeled after the stepwise annotation procedure proposed by Fuoli (2018). Outlining challenges in both *identifying* expressions for certain discourse functions and *classifying* them into precise categories, Fuoli (2018) underscores the reliability, replicability, and transparency of an annotation project. To this end, three overarching principles were proposed:

- **Principle 1**: All choices should be accounted for;
- **Principle 2**: The annotation guidelines should be tested and refined until maximum reliability is achieved; and
- **Principle 3**: Reliability should always be assessed, and reliability scores reported and discussed.

Table 4
Illustration of subtree suggester output.

Input text	Suggested Spans (all the subtrees)
My guess is that you would probably not believe in this approach yet.	[*My, My guess, guess, My guess is, is that you would probably not believe in this approach yet., My guess is that you would probably not believe in this approach yet., that, you, would, probably, not, that you would probably not believe, believe in this approach yet, that you would probably not believe in this approach yet, in, in this approach, this, this approach, approach, yet,*]

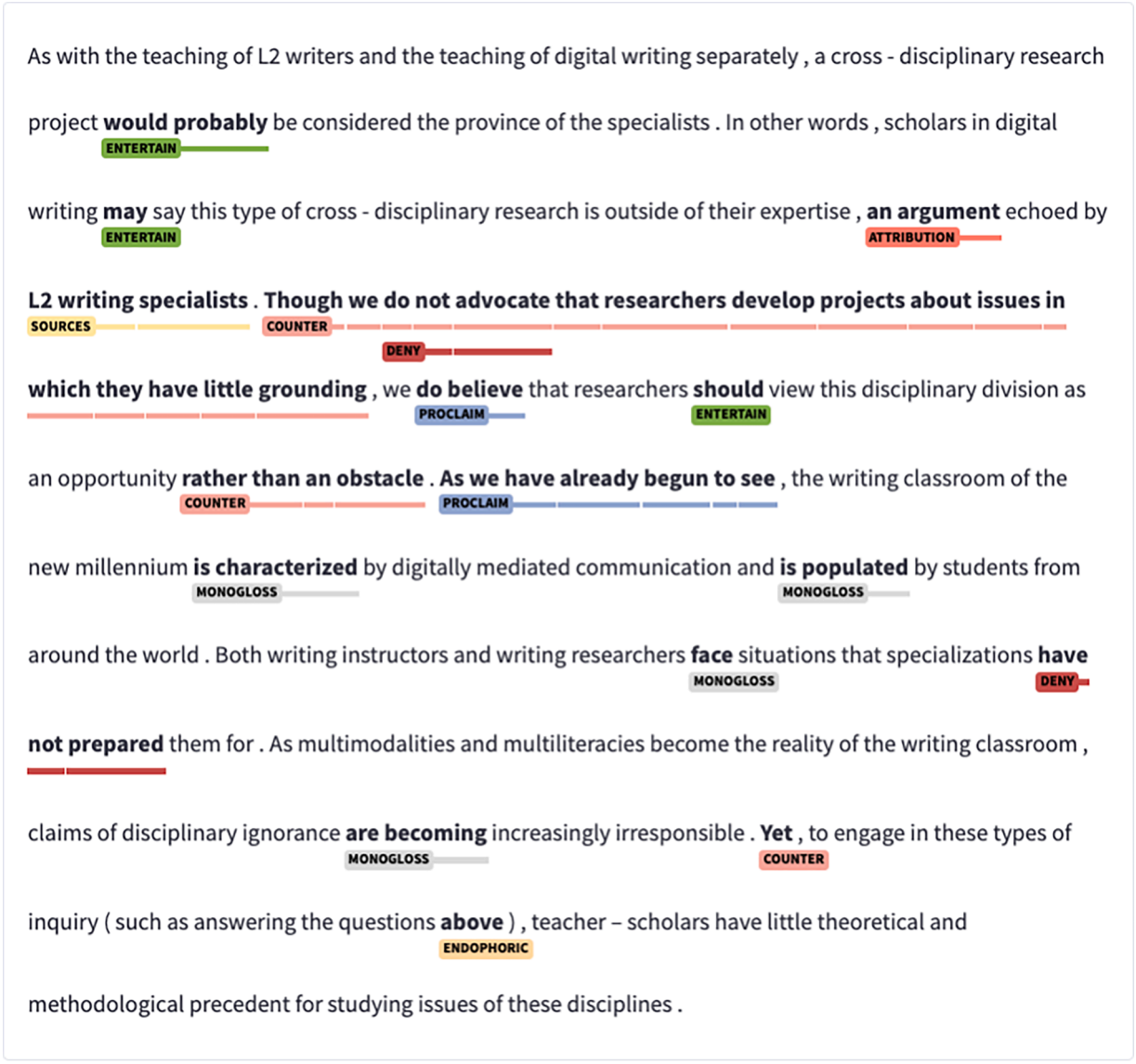

Fig. 4. Example prediction of the Engagement Analyzer.
Note. This excerpt was originally presented as Text C in Chang and Schleppegrell (2011) on page 147. This text was not part of the training data.

(Fuoli, 2018, pp. 246–7)

One identified challenge in annotating engagement resources includes the context-dependent nature of the linguistic phenomena. For example, an adverbial phrase, *according to X*, is known as an epistemic marker signaling the source of information of the authorial claim, but it does not mean that this expression is always used to accomplish that function. Similarly, grammatical resources such as negation are a typical Disclaim: Deny resource in engagement, but not all of them enact this function in discourse (see Fuoli, 2018, pp. 236–7). Such many-to-many relationships between the form and function must be reconciled during the ongoing annotation sessions and schemes should be updated and communicated to the user for transparency and replicability.

Step 1: Outlining the scopes and purposes of the tool

Before any of the work begins, it is important to have a specification of the NLP tool to develop (see Table 6 for the example). In the context of NLP for applied linguistics, the specification may contain information such as the target learner population, the target linguistic features, the target language use domain (TLU domain; Bachman & Palmer, 2010), and the kind of output that you desire, among others.

Step 2: Designing the NLP task

The second step is to design the NLP task from a technical perspective. Deciding on an appropriate NLP task for a given tool requires an understanding of common NLP tasks in terms of their inputs and outputs and common ML algorithms that you can use. For example, if the task requires each token in the input text to be tagged exclusively for one tag, an architecture used to perform POS tagging can be

Table 5
Overview of the steps in the tutorial on building a custom NLP tool.

Step	What to do	Goals	Guiding questions
1	Outlining the scopes and purposes of the tool	• Define the intended audience, learner population, target linguistic features, target language use domain, and desired output.	• Who will be using your tool for what purposes? • What kind of measurement do you need in your application? • What are the characteristics of text when you apply the tool for inferences? • Do your goals include identification, extraction, and/or classification of linguistic features, or something else?
2	Designing the task	• Choose an appropriate NLP task based on the input and output requirements. • Search for existing tag sets or annotation manuals, or create your own annotation scheme. • Define evaluation metrics for the model.	• What is the nature of the input (e.g., characteristics of text)? • What output does the tool produce—token-level tags, relationship extraction, or identification of (overlapping) spans?
3	Data sampling	• Sample annotation data from the target language use domain, representing a variety of proficiency levels and language backgrounds. • Determine the appropriate unit of analysis (sentence, paragraph, or whole text).	• What is the unit of analysis—Sentence, paragraph, or whole text? • What is the reasonable amount of data and batch size to complete annotation in a set amount of time?
4	Pilot annotation	• Pilot the annotation process using the drafted annotation manual and chosen software. • Verify that the annotated data can be converted to the format required for machine learning training.	• Are the annotation manuals clear enough to conduct the annotation? • Is the annotation tool configured properly to aid the manual annotation? • Are there already any "gold" standard examples of the target task or are you creating one from scratch?
5	Annotator training	• Recruit and train annotators, providing hands-on experience with the annotation guidelines and software.	• How long do the annotators need during the training? • Do they have enough background knowledge to conduct the annotation? If not, how would you assist their annotation?
6	Main Annotation	• Conduct the main annotation in batches, with regular meetings to resolve disagreements and refine the annotation guidelines. • Ensure annotation consistency through iterative reviews and adjudication.	• How should the accuracy of annotation be tracked? • How should the disagreed annotation be resolved?
7	Data preparation	• Preprocess the annotated data, convert it to the required format for machine learning, and split it into training, development, and test sets. • Address label imbalance through techniques such as oversampling.	• In what format do the data need to be? • How many datapoints do the annotated corpus contain for each category? • How do you address the imbalance in the data (e.g., annotating more data from minority category, oversampling already-annotated minority labels)?
8	Machine Learning Experiment	• Set a realistic goal for the model's performance based on human annotator agreement. • Obtain a baseline result using a simple model architecture. • Experiment with more complex architectures and hyperparameter tuning to improve performance.	• How many variants of the ML architecture are available for training? • What is the list of hyperparameters for each model and their ranges to compare the performance for?
9	Evaluation	• Evaluate the trained models using appropriate metrics such as precision, recall, F1 score, and Cohen's kappa. • Conduct cross-validation to assess the stability of the model's performance across different data splits.	• Which evaluation metrics are suitable? • What cross-validation methods do you employ or not?
11	Making inferences	• Write code to use the trained model for making predictions on new data.	• What information do you need to extract from the predictions to derive the desired linguistic measures?
10	Dissemination	• Share the trained model, evaluation metrics, and user manual through online repositories and academic publications. • Develop web applications or user-friendly interfaces to facilitate the use of the tool by non-technical audiences.	• In what way do you want to share your tool?

used. If one would like to identify a span and relationships between two spans, the coreference resolution task may be adapted. Researchers should conduct research to see whether there are any similar existing NLP tasks and even a gold-standard tag set that serves their purpose or informs their decisions (e.g., Penn Treebank Tagset for POS; Universal Dependency for dependency parsing). In some cases, the already-made annotation can provide a starting point for the development of new tasks (e.g., semantic role labels have been semiautomatically converted to argument-structure construction labels; Kyle & Sung, 2023). Sometimes, the expected output of the system may go beyond the scope of open-source packages, such as spaCy. In such case, one would need to engineer the architecture that can output the desired behavior using full-fledged ML libraries such as Tensorflow or Pytorch. Implementing such a complex ML architecture is out of the scope of the current tutorial.

At the conceptualization stage, it is also important to define the metrics to evaluate the model. Commonly used evaluation metrics for classification tasks include precision, recall, and F1 scores, but some specialized NLP tasks may require some tweaks in the evaluation metric. It is important, therefore, to plan the evaluation metrics to be used at this stage to ensure the successful evaluation of the results.

Table 6
Example specification of engagement analyzer.

	Specification
Intended Audience	L2 assessment researchers interested in writing and stance-taking; Writing instructors and their students; L2 writing instructors using genre-based pedagogy; Corpus linguists.
Learner population	English for Academic Purpose (EAP) students; Students in Content-Based Instruction (CBI) and Content and Language Integrated Learning (CLIL).
Target linguistic features	Engagement resource (as defined by Martin and White (2005)); Supplementarily meta-discourse features (as defined by Hyland (2005)).
Target TLU domain (or in-domain text)	English for Academic Purposes, particularly: - Disciplinary essays for university courses; - Essays written for EAP courses; - Timed argumentative writing (such as in TOEFL and IELTS).
Desired output	Identification of stance-taking expressions; Annotation of rhetorical functions based on Engagement framework (Martin & White, 2005); Instant visualization.

As mentioned already, the Engagement Analyzer was framed as a span categorization task (Eguchi & Kyle, 2023). Although the technical requirement was satisfied by searching for task that allows overlapping span representations (e.g., Papay et al., 2020) and available implementation through the spaCy package, it soon became clear that there were very few resources (e.g., annotation manual; gold-standard corpus) available that could be adopted. Therefore, an annotation guideline was drafted, drawing on the existing literature that discusses the framework (e.g., Chang & Schleppegrell, 2011; Lam & Crosthwaite, 2018; Lancaster, 2014; Wu, 2007; Xu, 2020). The resulting annotation guideline (after reflecting all the changes made during the main annotation) is accessible from the following webpage (https://egumasa.github.io/engagement-annotation-project/).

Step 3: Data sampling

The goals of this step are to sample the annotation data to maximize the balanced representation of the in-domain text and to decide the appropriate unit of analysis. First, it is important to represent a wide variety of text from the TLU domain. When training NLP models for the purposes of L2 analysis, it is arguably beneficial to represent a range of proficiency levels and language backgrounds (Kyle & Eguchi, 2024). Second, the appropriate unit of analysis should be decided given the nature of the NLP task. In an ideal circumstance, the entire corpus should be annotated; however, this is not always feasible due to resource constraints. A researcher has to make a decision on how to effectively sample the data widely from the in-domain text without compromising the quality of annotations.

Table 7 summarizes the source corpora, their characteristics, and target proportions in the annotation sample for Engagement Discourse Treebank (Eguchi & Kyle, 2023). Given the scope of the tool described in Step 1, the corpus data represents a variety of texts in the academic domain—written by both L1 and L2 writers for a variety of social purposes. The inclusion of argumentative essays in the dataset served two purposes—it allowed the researcher to represent the type of writing that is written as a part of standardized proficiency tests and the learner population with a wide range of proficiency levels. While the target proportion was not an absolute requirement for the final corpus, we used these numbers to conduct a stratified random sampling so that each annotation batch contains a balanced number of examples from each source corpus.

When sampling the data for annotation, it was decided to sample segments of texts, not the whole text, to strike a balance between the practical and substantive considerations. When the researchers plan to redistribute the annotated data to promote open-science practices, one obstacle would be the copyrights of the source corpora. In this study, only the BAWE (Alsop & Nesi, 2009) and MICUSP corpora (Römer & Swales, 2010) could be redistributed in its entirety. Therefore, it was determined that the dataset consists of three-sentence sequences randomly sampled from the entire source corpora. This approach allowed us to sample minimal contexts required for discourse annotation. It also allowed the annotated corpus to represent a wider variety of genres, styles, and stances than choosing to annotate only a handful of documents, which was considered essential in making a generalizable NLP tool.

Step 4: Pilot annotation

Once the annotation guideline is drafted, it is essential to pilot the annotation process. The pilot annotation aims to see if the data

Table 7
Source corpora of the EDT.

Corpus	General textual feature	Genres	Language background	Estimated proficiency levels	Target proportion
BAWE (Alsop & Nesi, 2009)	UK univ. assignments	Varying	L1 + L2	C1 +	0.35
MICUSP (Römer & Swales, 2010)	US univ. assignments	Varying	L1 + L2	C1 +	0.35
ICNALE (Ishikawa, 2018)	Timed essay	Argumentative	L2	A2 – B2	0.1
TOEFL 11 (Blanchard et al., 2013)	Exam response	Argumentative	L2	B1 – C1	0.1
CLC-FCE (Yannakoudakis et al., 2011)	Exam response	Argumentative/ letter writing	L2	B2	0.1

can be appropriately and efficiently annotated based on the annotation manual and to modify the annotation scheme as necessary. At this point, the researcher needs to decide on the software to help with the manual annotation and management of the project. A range of open-source and commercial software offers a graphical user interface to aid the intuitive manual annotation. For span annotation for Engagement Discourse Treebank (Eguchi & Kyle, 2023), an open-source Java-based application, WebAnno version 3.2 (Yimam et al., 2013), was selected (see Fig. 5). WebAnno helps the annotation workflow by visualizing the annotation sentence, allowing the user to annotate the span by dragging their cursors over the text span (in the annotation pane) and selecting a preset tag from a dropdown list (from the right panel).

Another goal of this step is to verify whether the annotated data can be converted to the data format for ML training. This is one of the most cumbersome but crucial steps because otherwise, the annotated data cannot be used to train the ML model. Even if the pilot annotation is tiny in size (e.g., 5–10 documents), it may be useful to verify the whole training and evaluation step before proceeding and see if the entire process runs without failing. See Step 7-9 for the details.

Step 5: Annotator training

The next step is to recruit and train annotators. In a supervised machine-learning project, the quality of the final model is hugely impacted by the reliability of gold-standard annotation. Therefore, it is ideal to ask experts who understand the target linguistic phenomena well to annotate the data. However, annotating a moderate amount of text for training machine-learning models (e.g., 100,000 tokens or more) is also important. Therefore, when a group of (non- or becoming-) linguists are hired, intensive training sessions should be planned with iterative feedback sessions.

Fig. 6 shows the summary of annotation sessions for the development of EDT. The entire annotation process was modeled after a stepwise annotation procedure proposed by Fuoli (2018). Two annotators were recruited from an upper-division linguistics course. They were both first-language speakers of English, and they had completed introductory linguistic courses, which covered functional syntax and semantics. However, since none of the two annotators took SFL-based functional linguistics courses, extensive hands-on training procedures were developed to familiarize them with the key concepts of SFL and grammatical terminologies used in the annotation manual. The training took approximately 10 weeks altogether (50–100 working hours).

During the training, it was also determined that the annotators needed additional assistance in correctly identifying clause boundaries (including main, subordinate, and embedding clauses) to make effective judgments in annotating engagement resources. Accordingly, an auxiliary task of clause segmentation was introduced (see Fig. 7). This clause boundary annotation was manually conducted until independent annotation was started in the fall of 2022. By this time, an automatic clause boundary detection model was developed based on the data during the training session and the initial four batches of main annotation (See Fig. 6), which reached satisfactory accuracy (F1 = 0.91). Thus, during independent annotation in Fall 2022 onwards, automated clause tags were provided in the distributed annotation data to speed up the process.

Step 6: Main Annotation

When it comes to developing an ML system, the goal of annotation is to construct a "gold-standard" annotation dataset, which provides consistent input-output relationships across the dataset. Therefore, time and effort should be spent on devising a procedure to improve the annotation quality iteratively. That is, it is necessary to revise the annotation manual so that the entire team of annotators will benefit from the incremental improvement. Reviews of the entire corpus should also be planned to ensure that the final guidelines are consistently applied throughout the annotated corpus.

The main annotation for EDT happened in three phases. First, to ensure a higher level of intercoder agreement, both annotators and the researcher blindly annotated the first 50 files from the corpus. The three versions were compared side-by-side, inconsistencies were resolved through discussion, and changes were reflected in the guideline. Any necessary adjustments to the annotation were made, and the adjudicated files were added to the annotated corpus. The second phase was a blind annotation of the same batch of data, followed by meetings to resolve any disagreements (Steps 4 and 5 in Fuoli's method; Fuoli, 2018). After each minibatch, the intercoder agreement was tracked progressively. This additional consensus-building process was repeated three times, totaling 150 files

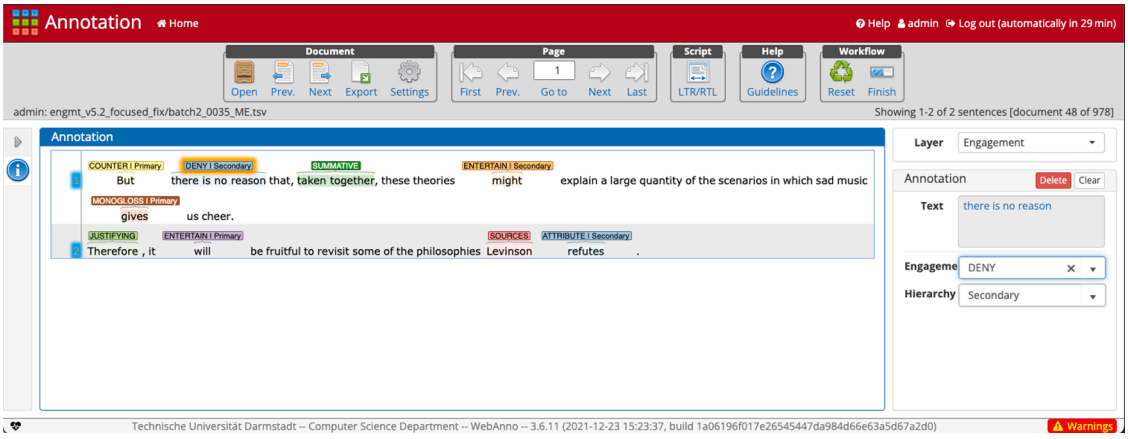

Fig. 5. Graphical User Interface of WebAnno annotation mode.

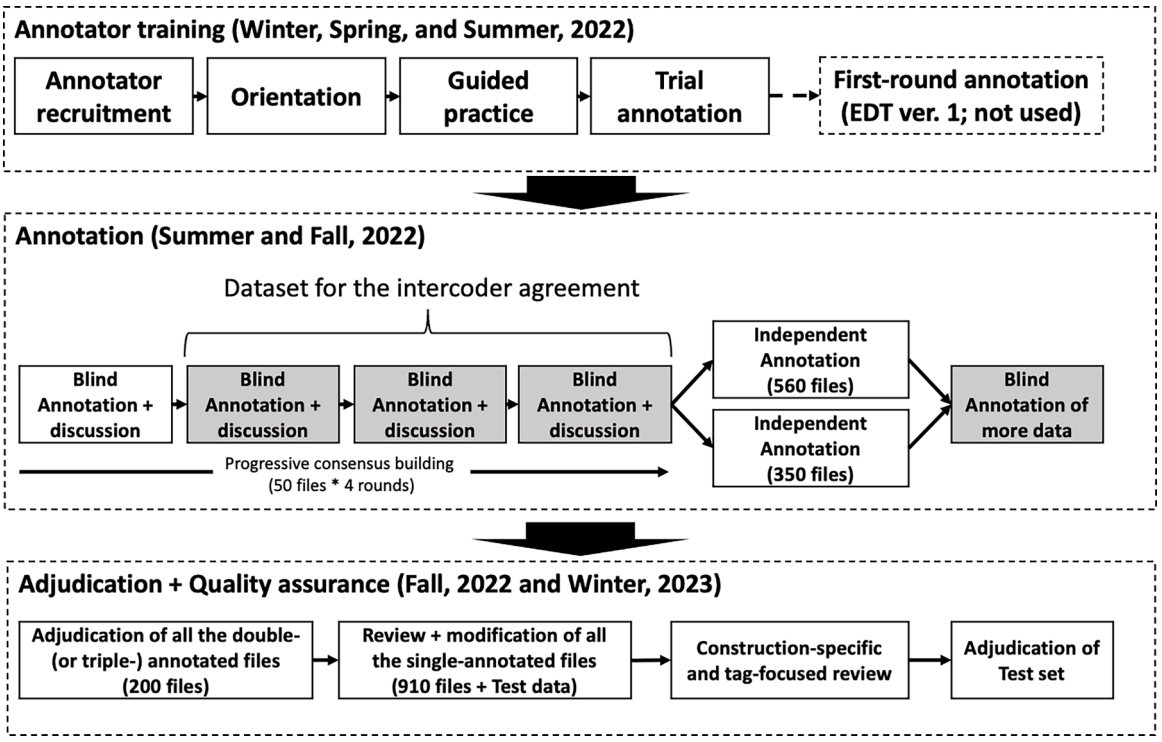

Fig. 6. Overview of the annotation procedure of the EDT.

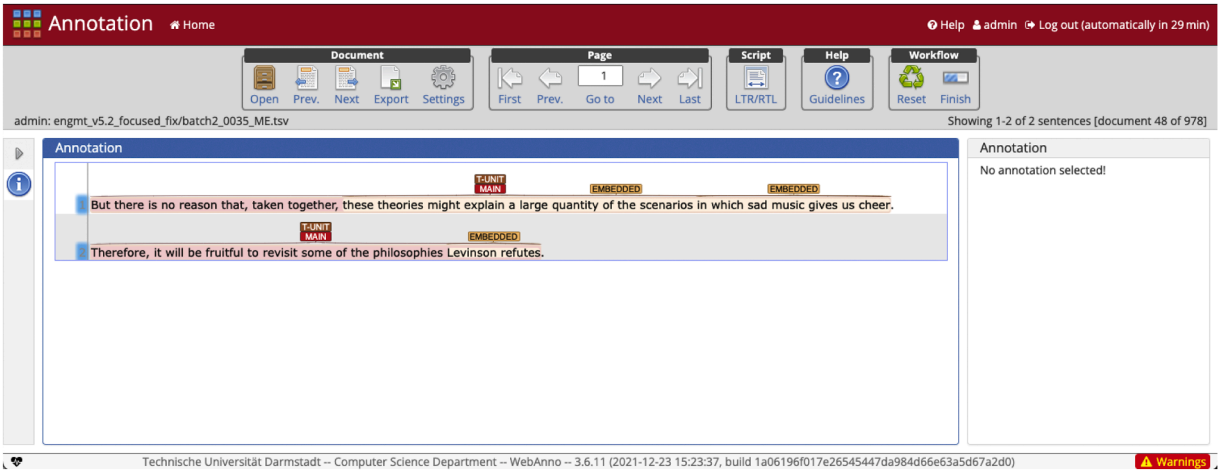

Fig. 7. Example of auxiliary clause boundary annotation.

independently coded by the annotators, followed by detailed discussions. The two versions of first-pass annotations for these 150 files were used to calculate intercoder agreement and used as a human-level annotation benchmark.

Additional steps were taken to ensure annotation consistency throughout the corpus. First, every single annotation file was reviewed and adjudicated by the first author (as the master annotator) for any deviation from the engagement framework. Other strategies were used to enhance the consistency of annotation, including (a) building annotated corpus databases, (b) indexing problematic examples, (c) a focused review of problematic examples, and (d) post hoc tag fixes through the concordance tool. The entire annotation team shared a single database, where each member could document any annotation decisions. This was to ensure the accountability of the annotation (Fuoli, 2018). The annotator documented their questions and possible alternative interpretations to discuss during the meetings. The database of annotated data also allowed the researcher to track any problematic cases. For example, when an annotator noticed any inconsistency in the annotation guidelines, such as the annotation span related to existential construction + negated nominal phrases (e.g., "*there is no reason*", "*there is no doubt*"), they were encouraged to create a new tag

categorizing the issue and index the annotation example in the database. Such indexing allowed the team to conduct comprehensive retrievals of any problematic cases throughout the corpus for discussion and review of the data for consistency. Such information was subsequently used for a focused review of inconsistent annotations throughout the corpus. To do so, the annotated data was converted to interlinear representation (e.g., WORD_POS+ENGAGEMENT-TAG), and AntConc concordance software version 4 (Anthony, 2022) was used to search for any consistent patterns.

Fig. 8 illustrates this process, where the search term "^there$_" returned all the occurrences of *there* in the annotated data. As can be seen, 124 occurrences were retrieved and sorted according to the tags. Among the topmost 19 hits in Fig. 8, a variety of tags were attached to existential constructions (the EX tag in the POS field)—two ATTRIBUTE, one ENTERTAIN, one EXEMPLIFYING, four MONOGLOSS, one PRONOUNCE, two TEXT-SEQUENCING, and eight empty tags (zeros). The sheer variety of tags is not a problem here. Indeed, we identified that existential construction can be used to ATTRIBUTE (e.g., line 2: "*there is much speculation*") or PRONOUNCE (e.g., line 9, "*there are no denying*"). However, we identified that there were inconsistencies in the spans of MONOGLOSS. For instance, lines 5–8 included existential *there* in the span of MONOGLOSS, while it was excluded in other cases (e.g., lines 12, 13, 15–17). During the adjudication, we focused on such inconsistencies and fixed them in our annotation. In this case, we clarified the span of MONOGLOSS in existential clauses (i.e., MONOGLOSS on a linking verb, excluding existential there), and fixed four files from lines 5–8 (see File ID in the File column).

Step 7: Data preparation

Once the researcher has completed the construction of gold-standard data, they can prepare the data for an ML experiment. This step involves (a) performing necessary pre-processing, (b) converting the data format for ML training, (c) making data splits, and (d) oversampling (if necessary), among others.

First, the raw annotated data and label may need to be pre-processed and verified for cleanness. This entails checking the misspelled gold-standard tags and verifying whether the gold-standard dataset is error-free.

Second, the annotated data should be converted to the format of data accepted in the specific ML training code. As mentioned in Step 4, this step is crucial because the packages for ML training accept certain pre-specified formats, and this format will depend on the task and the package you will be using for training. A commonly used format is vertical, as shown in Fig. 9. This data format, called CoNLL-U, is often used in NLP to represent multi-layered information about each token in a running text, including but not limited to Token ID, Wordform, Lemmatized form, Universal POS (UPOS), Penn POS tag (XPOS), Morphology, and Dependency information. CoNLL-U can include Semantic Role annotation and Named Entity Recognition (NER).

Third, after verifying the data format, the entire dataset is randomly split into three: training, development, and test sets. Usually, the ratio of the dataset size would be 80/10/10; however, the appropriate ratio needs to be justified on a case-by-case basis. In particular, when the researcher knows that there are minority labels that are difficult to sample using the 80/10/10 splits, a larger portion of the data could be sampled for dev and test set to secure a better representation of the minority cases in this dataset (for conservative estimates of the accuracy). Note that such a sampling strategy is only a temporary solution, and it is always better to increase the dataset size to obtain a better result. As discussed below, it may be helpful to devise a strategic annotation procedure to sidestep the problem of label imbalance rather than trying to fix the issue with a post-hoc approach.

Fourth, depending on the relative frequencies of labels in the dataset, it is necessary to correct the number of labels. Label imbalance is a frequently encountered problem in ML and NLP in particular (Aguiar et al., 2022). Although imbalances across the dataset do not undermine the quality of the corpus itself, they do potentially affect machine learning models during training. This is because models may perform better in the majority class than in the minority classes (X. Wang & Wang, 2022). Various techniques have been proposed in the ML literature to address this issue, including oversampling minority categories, under-sampling majority categories, and generating synthetic examples through data augmentation (for a review see Aguiar et al., 2022). To researchers trained in the psychometric approach to statistical modeling, the oversampling technique may appear to be data hacking, but this approach is a viable solution in the ML approach. Since (a) ML emphasizes prediction performance and (b) model training is conducted through multiple passes to the entire data with only a subset seen by the model at once, having the data points with minority categories repeated across the data helps mitigate the algorithm's biases towards the majority categories (with certain risks for the algorithm to overfit to the repeated instances to represent the label, resulting in potential under-generalization of the category).

In the Engagement Analyzer project, the following data preparation procedure was applied. Some of the steps were explained in the online tutorials due to its technicality and space limitations. First, the dataset is converted from the WebAnno output format to the one accepted for the spaCy span categorizer component. Fig. 10 illustrates this conversion step. It is important to note that not all annotation information is retained in the IOB format, thus not used to train the ML pipeline.

Second, the relative frequencies of labels were computed to see the potential imbalances and then split them into the Training, Development, and Test sets. Table 8 shows the frequency of each tag in the entire corpus and each held-out dataset. The EDT consists of 11,856 engagement annotations, with ENTERTAIN and MONOGLOSS being the most frequent categories. On the other hand, strategies—such as CONCUR, PRONOUNCE, and ENDORSE—accounted for only 1–2 % of the entire dataset. The issue of label imbalance is apparent as is often observed in the literature (e.g., Lam & Crosthwaite, 2018; Lancaster, 2014; Wu, 2007). To correct imbalances, an oversampling approach was chosen because there was no reliable model to annotate or create synthetic examples, and the dataset was too small to discard majority labels. In Eguchi and Kyle (2023), we simply duplicated annotation files containing minority labels for each of the training and development sets. Oversampling should be applied after the data split is made to prevent data leaks (i.e., the issue of same files appearing in both the training and development). Table 9 presents the number of labels after oversampling. Note that the differences in frequencies of minority and majority cases were narrowed. It is worth noting that in a span categorization dataset such as this, perfectly balanced sampling is impossible because the labels are not independent (oversampling documents for a specific label inevitably increases other categories included in the same document).

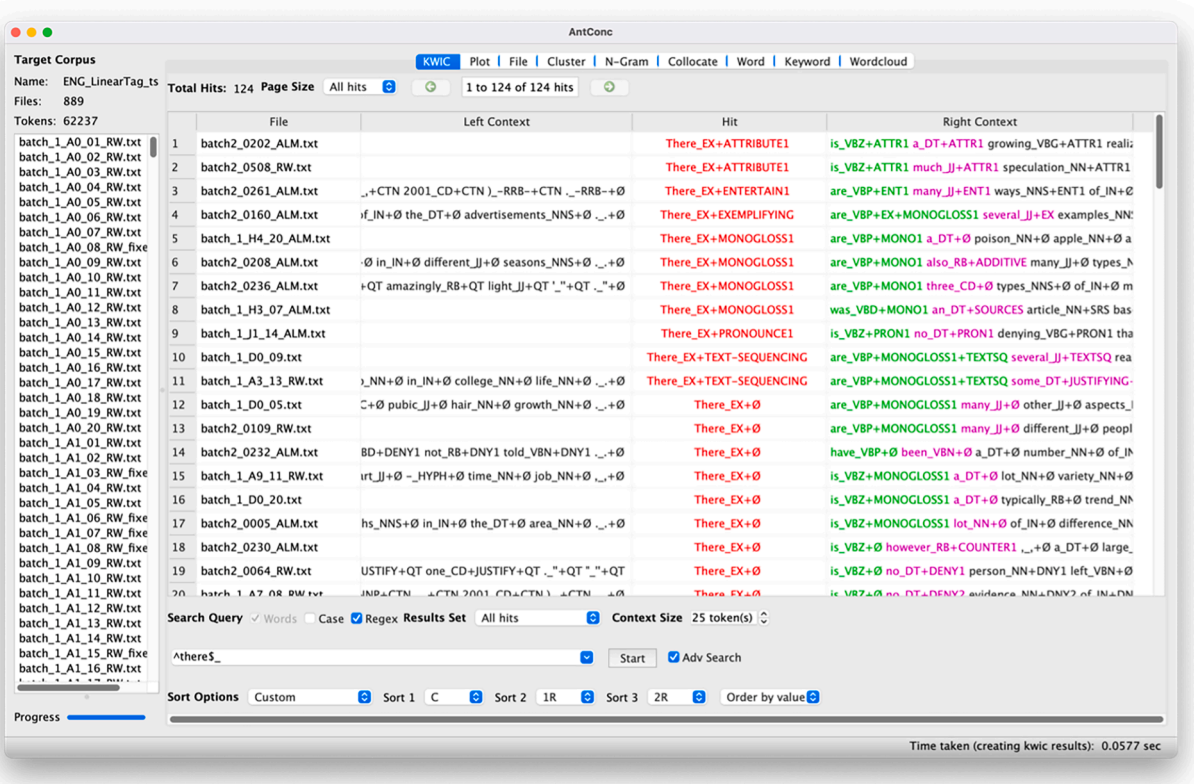

Fig. 8. Illustrative AntConc display for tag-specific adjudication.

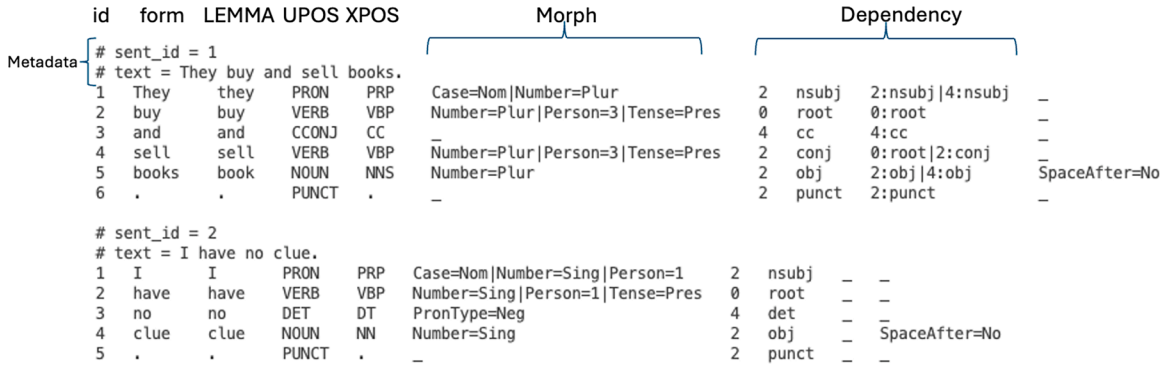

Fig. 9. CoNNL-U data format for common NLP pipeline.
Note. In CoNNL-U format, information about each token is represented in a row. For example, the word form "they" (i.e., the first token in sent_id of 1) is a (personal) pronoun (PRON/PRP), in a Nominal case and Plural in number. The dependency information shows that this token is a nominal subject (nsubj), and its dependency head is the second token of the sentence (i.e., buy).

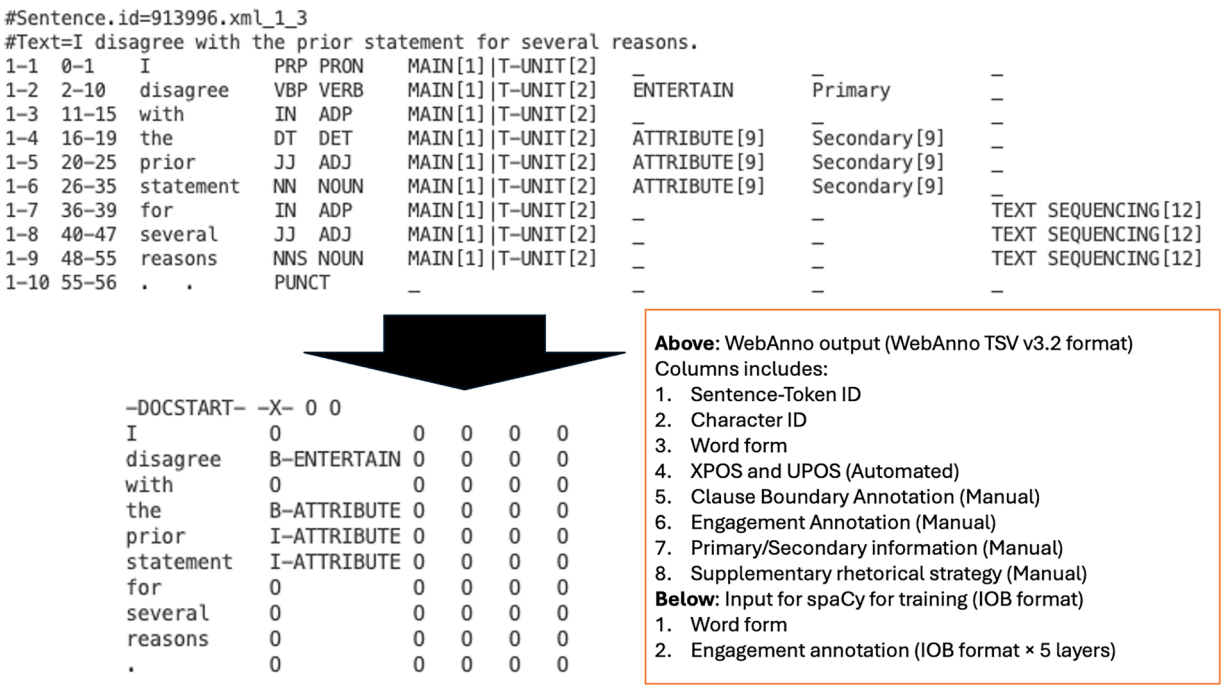

Fig. 10. The illustration of data conversion from WebAnno tsv output to the IOB format.

Step 8: Machine Learning Experiment

Once the data annotation, curation, and preprocessing have been completed, one can begin the machine-learning experiment. Although much of the technical aspects must be delegated for space constraints, here are some general steps in training machine-learning models. For the ML experiment, the spaCy package version 3.4.4 was used. The Google Colabnotebook will show you steps 8b and 8c.

Step 8a: Setting the goal for the ML experiment

The accuracy of the supervised machine-learning projects depends on the complexity of the task. When the researchers develop a new task, it is useful to know how well a human annotator performs it. The idea is to assess the accuracy of the automated tagging model relative to the average human annotator agreement as a good starting point. Table 10 presents the inter-coder agreement between the two human coders. From these figures, it is expected that a realistic expectation for the accuracy of the machine learning model on this task would be around 0.70 (given the inter-coder agreement of 0.67 in both Cohen's Kappa and macro F1 scores). The by-tag agreement between the human coders implies that the ML models would struggle to identify categories such as ATTRIBUTION, ENDOPHORIC, PROCLAIM, and SOURCES.

Step 8b: Training the baseline model(s)

Table 8
Numbers of unique tags in the original corpus.

	Training	Dev	Test	Total	Percentage
Deny:	738	67	82	887	7.481 %
Counter:	827	107	112	1046	8.823 %
Concur:	104	11	12	127	1.071 %
Pronounce:	259	31	28	318	2.682 %
Endorse:	122	10	15	147	1.240 %
Entertain:	2280	269	288	2837	23.929 %
Attribute:	887	105	108	1100	9.278 %
Monogloss:	2211	257	274	2742	23.128 %
Citation:	482	68	68	618	5.213 %
Sources:	707	69	79	855	7.212 %
Endophoric:	162	26	25	213	1.797 %
Justifying:	795	84	87	966	8.148 %
Tag count	9574	1104	1178	11,856	

Table 9
Numbers of unique tags in the oversampled corpus.

	Training set	Dev set	Test set	Total	Percentage
Deny:	3016	235	327	3578	6.726 %
Counter:	3471	437	598	4506	8.471 %
Concur:	984	80	116	1180	2.218 %
Pronounce:	2363	240	255	2858	5.373 %
Endorse:	1256	113	191	1560	2.933 %
Entertain:	8219	998	1216	10,433	19.613 %
Attribute:	5238	583	655	6476	12.174 %
Monogloss:	6995	734	832	8561	16.094 %
Citation:	2241	352	379	2972	5.587 %
Sources:	4385	393	502	5280	9.926 %
Endophoric:	1650	224	266	2140	4.023 %
Justifying:	3005	324	321	3650	6.862 %
	42,823	4713	5658	53,194	

Table 10
Intercoder agreement (using Annotator B as reference).

	Benchmark by Read and Carroll (2012)	Precision	Recall	F1-score	Data points (k)
Attribution	.379	0.65	0.55	0.6	264
Citation	n/a	0.97	0.93	0.95	203
Counter	.603	0.77	0.95	0.85	231
Deny	.451	0.88	0.86	0.87	232
Endophoric	n/a	0.66	0.58	0.62	89
Entertain	.459	0.87	0.79	0.83	747
Justifying	n/a	0.83	0.81	0.82	557
Monogloss	n/a	0.8	0.82	0.81	861
Proclaim	.336	0.32	0.55	0.4	78
Sources	n/a	0.62	0.52	0.57	152
_		0	0	0	315
Accuracy				0.72	3729
Cohen's kappa				0.67	
Macro Avg		0.67	0.67	0.67	3729
Weighted Avg		0.73	0.72	0.72	3729

Note. Benchmark is reported based on a previous study by Read and Carroll (2012). The granularity of the tags was different between the studies. n/a indicates no corresponding tags were investigated in Read and Carroll.

After establishing the human benchmark, we can train a model (see online tutorials for actual implementations; https://github.com/egumasa/engagement-analyzer-train). Recall from Fig. 3 that the spancat component consists of three pipelines connected back-to-back (token embedder, span candidate suggester, and span categorizer). The codes to train the baseline models have already been implemented in the spaCy library. We must follow the recommended steps: prepare the data (see Fig. 10), edit the model configuration, and enter the command line to run the training. The online tutorial shows how to conduct these steps for two baseline models. The two models differ in the token embedding methods (the first component in Fig. 3)—The first is a multi-hash embedding with a Convolutional Neural Network (CNN), which is essentially the same as the *en_core_web_lg* model in spaCy package; The second model uses a Transformer-based token embedder (similar to *en_core_web_trf* model). The first architecture should be trained relatively

quickly, while the second architecture may require a few hours to train on Colaboratory using GPU resources. Once each model is trained, another command line could be used to evaluate the trained model (see Step 9 for details). At this point, it is recommended to train default models by spaCy to obtain the baseline results. When the result is unsatisfactory or if there is room for improvement—such as recall is significantly lower than the precision, or vice versa—one should consider changing the model configurations, either by trying different architecture (Step 8c) or by tuning the hyperparameter (Step 8d), or both.

Step 8c: Training models with alternative architectures (Optional)

Proposing alternative architectures essentially entails engineering machine-learning models (also known as architecture engineering). Architectural changes that work well within and across the context potentially make a substantive contribution to the field of machine learning. As such, this will require both foundational and advanced knowledge of the machine-learning approach to NLP and practical coding skills (including the PyTorch library and the Thinc library that underlies the spaCy machine-learning component).

In Eguchi and Kyle (2023), two additional model architectures were tested. The first architectural change concerns the addition of the Bidirectional Long-Short Term Memory (Bi-LSTM) layer on top of the Transformer embedder. This model attempts to learn additional sequential information that is useful to disambiguate the rhetorical function of a span using the surrounding token information. The second architecture tested used two separate transformer embedders, which learn the latent representations for the candidate span separately. These two representations were then added together before making the final prediction. This arguably improves the prediction by complementing the weaknesses of each model. More rationales regarding the model architecture are presented in Eguchi and Kyle (2023).

Step 8d: Hyperparameter tuning (Optional)

When training a model, the same general model architecture can be implemented with slightly different configurations (i.e., *hyperparameters*), such as the size of the hidden dimensions, the number of non-linear activation function layers, and the learning rates to update the parameter weights. This can result in differential results even within the same model architecture. Searching for the best configuration for each model architecture is called hyperparameter tuning.

There are several approaches to conducting hyperparameter tuning, including grid search, random search, and Bayesian search. In grid search, the researcher specifies a set of discrete values for each hyperparameter and tries all the possible combinations of these hyperparameter spaces, which are subsequently selected for the best results. A random search can be conducted by letting a computer select a set of hyperparameter settings within pre-specified spaces (either discrete or continuous) and iterate this procedure for a given number of times. The Bayesian search tries to improve on the random search by having the algorithm change the hyperparameter value that might influence the result more over the iterations. Hyperparameter tuning may be less impactful or substantively important than the model architecture, but certain hyperparameters can improve the overall prediction to some extent. Eguchi and Kyle (2023) conducted Bayesian search methods to create a total of 205 models, eight of which were then considered for a 5-fold CV.

Step 9: Evaluation

For a classification model such as span categorizer, Precision, Recall, and F1 scores are commonly used (see Preliminary section for the details of this family of evaluation metrics). Precision, Recall, and F1 scores are normally calculated per label, allowing for the finer-grained evaluation. To obtain an overall metric for evaluation, two averaging methods can be used for different purposes—Macro

Table 11
Results of 5-fold cross-validations.

	5-fold CV (on held-out Test set)											
	Macro P			Macro R			Macro F1			Kappa		
Architectures	*M*	*Min*	*Max*	*M*	*Min*	*Max*	*M*	*Min*	*Max*	*M*	*Min*	*Max*
Transformer baseline												
(a) RoBERTa-base + Maxout (spaCy's default)	0.745	0.728	0.764	0.695	0.672	0.716	0.715	0.695	0.732	0.661	0.651	0.697
(b) RoBERTa-Academic + Mish	0.729	0.715	0.738	0.675	0.632	0.717	0.695	0.665	0.719	0.647	0.602	0.674
Transformer + LSTM												
(c) RoBERTa-base + LSTM + Mish two-way	**0.754**	0.734	**0.772**	0.710	0.670	0.734	**0.728**	0.696	**0.750**	**0.689**	0.634	**0.712**
(d) RoBERTa-Academic + LSTM + Mish two-way	0.752	**0.741**	0.763	0.691	0.667	0.715	0.715	0.696	0.726	0.678	0.648	0.682
(e) RoBERTa-Academic + LSTM + Mish two-way	0.743	0.710	0.763	0.704	0.666	0.726	0.719	0.696	0.736	0.674	0.643	0.697
(f) RoBERTa-Academic + LSTM + Mish two-way	0.747	0.735	0.770	0.695	0.679	0.719	0.716	**0.706**	0.725	0.665	**0.658**	0.686
Dual Transformer + LSTM												
(g) RoBERTa-Academic + Mish-two-way * 2	0.741	0.723	0.756	0.708	0.681	**0.737**	0.721	0.705	0.745	0.658	0.654	0.708
(h) RoBERTa-Academic + Mish-two-way * 2	0.718	0.691	0.744	**0.724**	**0.702**	0.733	0.718	**0.706**	0.732	0.664	0.635	0.692

Note. The columns showing the mean of each metric are highlighted. Bold faces indicate the highest score from each column. For each of the three architectures, several models were tested with varying hyperparameter settings. Online supplementary material provides the exact hyperparameter settings for each of the eight models.

and Micro averages. Macro average is a simple average—i.e., the sum of by-category scores divided by the number of categories. This results in equal weights across the categories in the calculation. As such, this is desirable when the researcher cares about the performance of minority categories. Micro averages are calculated at the item level regardless of the category boundary. Micro averages could be a more direct reflection of the model performance when they are used in inferences, as long as the relative sizes of the categories are similar to those in production.

To evaluate the Engagement Analyzer, macro averages were used to give equal consideration to majority and minority categories. For the highest-performing eight models from the hyperparameter search, five-fold CVs were conducted to see whether the selected models resulted in stable performance across different folds of data. Table 11 presents the result of 5-fold CVs for each. It shows the mean, minimum, and maximum scores for each of the metrics considered. Some variability in performance was observed across models within the same architecture, highlighting that hyperparameters (i.e., the size of the hidden parameters, the type of non-linear activation function) had some impacts on the model performances (See online supplementary material for exact configuration for each model). First, the transformer+LSTM architecture (Models c–f with different hyperparameters) tends to score highest among the three architectures, particularly on Precision, F1, and Kappa coefficient. Second, the dual-transformer models (Model g and h with different hyperparameters) scored highest on Recall. An additional benefit of the dual-transformer would be that the ranges of F1 scores across 5-fold CV tended to be narrower, particularly in terms of Minimum, although no statistical comparison was being made due to the small number of folds being tested (i.e., 5-fold). Finally, Model (c) in Table 11 was identified as the best-performing model configuration.

Step 10: Using the trained model to making predictions

After completing training, every model can make inferences, although their performances vary. To obtain the inference results, one needs to write additional codes, which, in this case, are in the Python language. The online tutorial shows the steps to make inferences with the trained spaCy spancat models.

Step 11: Dissemination

At the end of the project, researchers can share the outcome of the experiment—the evaluation metrics and the resulting machine-learning model(s). The evaluation metrics can be reported to the research community in a paper, as in Eguchi and Kyle (2023). In addition to the academic output, researchers should share the model with the research community to promote open science. This can be accomplished in several steps. One can share the trained model through online repositories such as Huggingface or GitHub. As of August 2024, Huggingface offers free storage for relatively big models. Through this platform, researchers can share trained models so that anyone can download and use them for their own purposes. Another strategy for dissemination is to make a linguistic analysis tool that uses the trained model in order to facilitate its applications by non-technical audiences. One possible form is to develop a web application where the user can input their text and make an inference with the trained model as in the Engagement Analyzer demo app hosted on Huggingface Space (https://huggingface.co/spaces/egumasa/engagement-analyzer-demo).

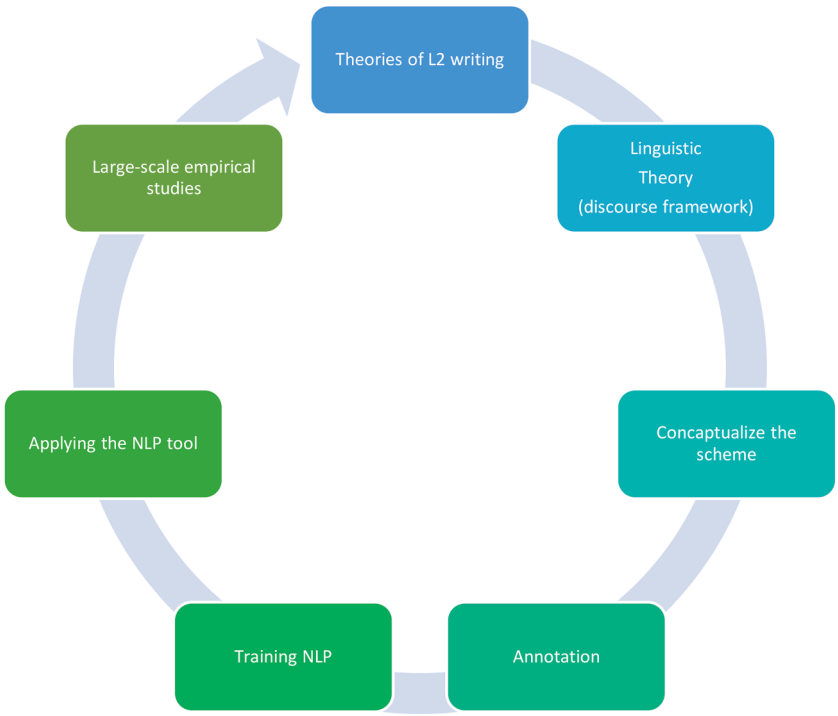

Fig. 11. Integrating a custom NLP pipeline in the L2 writing research cycle.

Discussion, implications, and practical considerations

Implications for custom NLP tools in applied linguistics

As the existing literature has demonstrated, automating annotation of various theoretically important linguistic features contributes to scaling empirical investigation on the topic, thereby facilitating theory-building and application in practice (e.g., Biber et al., 2004; Chen & Meurers, 2016; Crossley et al., 2019; Graesser et al., 2004; Kyle, 2016; Lu, 2011, 2012; Yoon, 2017). With the breakthrough innovations in NLP methods (i.e., pre-trained Transformer models), developing custom NLP tools to serve wider purposes in applied linguistic research is becoming increasingly possible. Beyond the Engagement system framework, one could build a tool by incorporating and/or expanding on already existing annotation frameworks and annotated corpora, such as Rhetorical Structure Theory (Carlson et al., 2001; for a review see Hou et al., 2020) and Move/Step analysis (Cotos & Pendar, 2015; Swales, 1990). Research could also investigate the relative benefits of identifying linguistic phenomena through traditional rule-based pattern matching vs. probabilistic approaches for identifying metadiscourse features (Hyland, 2005) and register-based corpus features (Biber, 1988; Biber et al., 2004). The probabilistic approach is arguably beneficial as it can effectively deal with the lexical items that are out-of-vocabulary in any of the existing lists. Furthermore, an emerging line of research, including the current work on the Engagement framework, attempts to translate existing linguistic theories into coherent annotation schemes for identifying important features in L2 research—including Goldbergian Verb Argument Structure Constructions (Kyle & Sung, 2023; Sung & Kyle, 2024). We believe that interdisciplinary collaboration between applied linguists and NLP engineers will push the boundary of the next-generation corpus tools to help theory building as well as practical application at the same time (see Meurers & Dickinson, 2017).

Fig. 11 presents a schematic representation of the interdisciplinary research process (see also Norris & Ortega, 2003). First, researchers attempt to identify important features of L2 writing to consider in their research. This often informs the choice of linguistic theory based on which they need to operationalize the "quality" of writing (i.e., Engagement in our demonstration). Subsequently, if the researcher decides to build a custom NLP tool for this purpose, they will translate the conceptual framework into a concrete annotation guideline and annotate corpora from the chosen TLU domain. Once they obtain reliable annotation, they train an NLP tool and evaluate their model against some benchmark to judge their quality (e.g., the inter-coder reliability). In the final two steps, the researchers can apply the NLP tools to empirical studies to identify the desired linguistic feature and conduct large-scale studies looking at proficiency effects and/or longitudinal development. This hopefully contributes to the understanding of how the target writing features develop under different conditions, which may, in turn, inform how these features should be looked at in future research, initiating the second iteration of the entire research cycle.

Important practical considerations

When a researcher wishes to develop their own NLP tool, it is useful to consider the following.

First, one must think about whether there is really a need to develop a new machine learning algorithm to annotate the linguistic feature of interest—that is to say, whether there is any existing tool to obtain a "sufficient" proxy of the intended linguistic features. When the researcher finally decides to take a machine-learning approach, perhaps one pertinent question includes the size of the annotated dataset to train a usable model in practice. Unfortunately, unlike statistical power analyses on simple general linear models (e.g., *t*-test, ANOVA), there is not a simple answer to the number of data points or the size of the corpus that is required to obtain reliability. The accuracy of the trained model depends on the complexity of the NLP task (e.g., grammatical tagging, discourse features, pragmatic features), the range of linguistic features that realize the functional categories, and how well the pre-trained language model captures the latent distributions useful in distinguishing these features.

Indeed, strategies to deal with the lack of labeled data have been the active area of research. Recently, the availability of an open-source pre-trained (large) language model significantly alleviated the necessity of large-scale labeled data. As demonstrated through the current case study, a reasonable accuracy of annotation (comparable to the human benchmark) can be obtained with a total of 11,856 labeled spans across approximately 100,000 tokens. While obtaining this amount of annotated data in a single study is still challenging, collaborative effort in the field will make such a study possible.

Another important consideration is the choice of appropriate machine-learning models. While the current tutorial focused on a supervised machine-learning approach, the application of generative models such as GPT to linguistic annotations is quickly gaining popularity (Mizumoto et al., 2024; S. Wang et al., 2023). For example, Kim and Lu (2024) demonstrated that GPT-3.5 can be used to obtain F1 of 0.8 or above in move-step annotations after fine-tuning it with 80 training documents amounting 1556 sentences. Considering such an emerging line of research demonstrating the capacities of LLMs for various linguistic tasks (Brown et al., 2020), an effective strategy could be to first try to annotate the target linguistic phenomena with LLMs through zero- or few-shot prompting techniques (as in Kim & Lu, 2024) and observe what challenges these generative models may face (or not). This information may allow one to plan their research—what machine-learning approaches they need, which specific category can be challenging for models, etc. Best practices on using LLMs for linguistic annotation, however, merit further investigation as a field and thus are out of the scope of the current tutorial.

Finally, reproducibility and replicability have been some of the main concerns in science in general. Research using NLP tools, particularly those using deep learning methods, is no exception. Although one main goal of introducing these tools is to minimize subjectivity, it is important to mention that reproducibility is ensured when the exact same model is used in the analyses. When different versions of the same model or architecture are used, they will produce different results. For this reason, when it comes to using the NLP models in research, the researchers should be encouraged to report the exact version of the model being used and

possibly their specifications.

Implications for the engagement system

The NLP approach to annotating discourse features, such as the current demonstration, raises a series of important questions in translating the systemic network of engagement (Martin & White) into a coherent description of each category for consistent annotation throughout the corpus. Our intercoder reliability (Cohen's Kappa of 0.67) substantially improved upon previous work (e.g., Read & Carroll, 2012); however, it still implies multiple challenges in identifying rhetorical strategies and their lexico-grammatical realizations. This finding is reminiscent of Fuoli's (2018) claim that the descriptions and definitions in the extant literature on Engagement may be insufficient to categorize specific instances in discourse. While we hope that the application of Fuoli's stepwise annotation procedure ensured to maximize the reliability between two annotators, the study indeed re-discovered challenges in distinguishing some categories of Engagement—ENTERTAIN and PROCLAIM, MONOGLOSS and ENTERTAIN, and MONOGLOSS and ATTRIBUTION.

One of the most problematic is the category of PRONOUNCE. Martin and White (2005) describe PRONOUNCE as follows:

The category of pronounce covers formulations which involve authorial emphases or explicit authorial interventions or interpolations. For example: *I contend…, The facts of the matter are that…, The truth of the matter is* that…, *We can only conclude* that…, *You must agree* that.., intensifiers with clausal scope such as really, indeed, etc. and, in speech, appropriately placed stress (e.g., The level of tolerance IS the result of government intervention). (Martin & White, 2005, p. 127)

What is significant from this description is that PRONOUNCE has to do with utterances involving "authorial emphases, interventions or interpolations" (p.127). A few other examples that Martin and White (2005) use to illustrate the idea of PRONOUNCE include:

- It is absolutely clear to me that…
- We have to remember that…
- What *really* differentiates cool from worm couples is…

One of the struggles the annotation team faced was the treatment of commonly occurring clausal expressions in academic discourse such as "*it is important/essential/evident*" (called extrapolated that-clauses followed by an adjectival complement ; Biber et al., 1999, 2021). In the SFL tradition, these extrapolated that-clause constructions are often considered a means to express evaluation objectively (Halliday & Matthiessen, 2014, p. 688; Martin & White, 2005) by hiding the agentive role of the writer in the immediate context—or interpersonal metaphor (Halliday & Matthiessen, 2014). After discussion and review of the literature on evaluative language, we decided that the categorization of extrapolated *that*-clauses would primarily be motivated by the lexical semantics (and by extension, functional potentials) of the adjectival complements which control the *that*-clauses. For example, when the controlling adjectives are "*important*" or "*essential*", there is good reason to treat them under MONOGLOSS. Based on the dialogic potential of the expressions "*it is important*" or "*it is inevitable*", these would be considered as undialogized utterances (ignoring the dialogic potential in the immediate discourse). This undialogized nature of utterances is supported by the fact that one can add ENTERTAIN resources to show recognition of this dialogic nature of discourse (e.g., *It seems important; It can be essential*). Another reason is that the extrapolated that-clauses in "it is important" and "it is essential" apparently lack the author's explicit commitment to the proposition compared to prototypical examples of PRONOUNCE, such as *I contend* or *I conclude*. Thus, a seemingly un-dialogized, non-explicit comment adjective (*important, critical, necessary*) would fit well with the prototype description of MONOGLOSS in Martin and White (2005).

On the other hand, variants of extrapolated *that*-clause constructions can be categorized under PRONOUNCE when the adjectives reveal the writer's epistemic commitment to the proposition introduced in the that-clause (e.g., *it is clear* and *it is evident*). This decision would also be supported by the fact that their derived adverbial forms are introduced as prototypical PRONOUNCE-enacting resources ("clearly" and "evidently"; Martin & White, 2005). To the best of our knowledge, there is at least one previous research example in the Engagement literature that included "it is evident" as an example of PRONOUNCE (Chang & Schleppegrell, 2011), although we were not able to identify mentions of this particular construction in the original Martin and White (2005) monograph. Such extrapolation of the original Martin and White (2005) volume may be debatable among SFL analysts, but a comprehensive analysis of this construction is beyond the scope of the current study.

In essence, the decision-making process above illustrates not only the challenges but also the possible benefits for theory building. That is, the process of annotation (although to train an NLP model) encourages us to consider each instance in our data according to the linguistic theory on which the annotation project draws. We concur with Fuoli (2018) that each annotation decision should be made accountable, and through this process, we may uncover instances that contradict the current theory and offer some alternative classification. This process, often demonstrated in the literature on engagement (Lancaster, 2014; Lee, 2010, 2017), can in turn refine the annotation scheme, potentially contributing to the development of theory.

Conclusion

In this tutorial, we aimed to showcase the process of developing a custom NLP model, from the task design to corpus annotation, model training, and evaluation. We provided a step-by-step tutorial based on the development and validation of Engagement Analyzer, a span categorization model trained with the spaCy library in Python. We also discussed the implication of the discourse annotation for machine learning to the theory refinement and important practical considerations. The online tutorial was also provided, where the

readers can reproduce the machine-learning experiment discussed in the tutorial.

Suggested readings

The suggested readings are intended to help students and researchers gain the necessary knowledge and skills, ranging from conducting linguistic analyses using existing NLP tools to understanding how machine learning works and designing alternative architectures for models.

- Introduction to Natural Language Processing:
  - Jurafsky and Martin (2009) provides a classic introduction to the field of Natural Language Processing. It covers definitions of a wide range of NLP tasks, and the algorithms often used to solve them. The most recent edition (in preparation) is freely accessible through their website.
- Introduction to modern machine-learning approaches to NLP:
  - Hagiwara (2022) offers an accessible introduction to the key building blocks of modern NLP architectures and their practical applications. The book begins with foundational concepts in machine learning and NLP and then introduces important architectures such as Recurrent Neural Networks, Convolutional Neural Networks, and the Transformer architecture. The code examples are implemented using the AllenNLP library in Python, but the conceptual explanations provide a valuable resource for researchers.
  - Tunstall et al. (2022) provide both conceptual and practical introductions to the Transformer architecture. While the practical examples (i.e., implementation code) rely on the Huggingface ecosystem and the PyTorch library, the conceptual discussions (particularly Chapter 3, "Transformer Anatomy," and Chapter 9, "Dealing with Few to No Labels") are excellent resources for learning about the development of NLP tools.
- Practical guides to the spaCy library:
  - The official spaCy documentation provides a comprehensive resource for understanding and using the spaCy library.
  - Altinok (2021) provides a comprehensive series of practical tutorials on using the spaCy library (version 3), including POS tagging, Dependency Parsing, Named Entity Recognition, and model training.

More resources are provided in the online tutorial.

CRediT authorship contribution statement

Masaki Eguchi: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Kristopher Kyle:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We thank the editor-in-chief, Professor Shaofeng Li, special issue editor, Professor Peter Crosthwaite, and two anonymous reviewers for their insightful comments. We express our sincere gratitude to two undergraduate annotators, Aaron Miller and Ryan Walker, for their meticulous work on the discourse annotation that made this work possible. This project was supported by the following grants/awards: the Duolingo English Test's Doctoral Dissertation Award 2022, the International Research Foundation for English Language Education (TIRF) Doctoral Dissertation Grant 2022, the National Federation of Modern Language Teachers Association and the Modern Language Journal (NFMLTA-MLJ) Dissertation Writing Support Grant 2022, the Graduate Student Research Award at the Linguistics Department, University of Oregon, and Kristopher Kyle's institutional research funds. Part of this project was also supported by JSPS KAKENHI Grant Number JP24K16141 awarded to Masaki Eguchi. During the preparation of this work, the author(s) used generative AI tools (i.e., GPT-4o and Claude Sonnet 3.5) to proofread parts of the manuscript for grammatical correctness, clarity, and coherence. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

References

Aguiar, G., Krawczyk, B., & Cano, A. (2022). A survey on learning from imbalanced data streams: Taxonomy, challenges, empirical study, and reproducible experimental framework. *arXiv:2204.03719*. http://arxiv.org/abs/2204.03719.

Alsop, S., & Nesi, H. (2009). Issues in the development of the British academic written English (BAWE) corpus. *Corpora, 4*(1), 71–83. https://doi.org/10.3366/E1749503209000227

Altinok, D. (2021). *Mastering spaCy: An end-to-end practical guide to implementing NLP applications using the python ecosystem*. Packt Publishing.

Anthony, L. (2022). AntConc (Version 4.1.1) [Computer software].

Bachman, L. F., & Palmer, A. S. (2010). *Language assessment in practice: Developing language assessments and justifying their use in the real world*. Oxford University Press.
Bax, S., Nakatsuhara, F., & Waller, D. (2019). Researching L2 writers' use of metadiscourse markers at intermediate and advanced levels. *System, 83*, 79–95. https://doi.org/10.1016/j.system.2019.02.010
Bestgen, Y., & Granger, S. (2018). Tracking L2 writers' phraseological development using collgrams: Evidence from a longitudinal EFL corpus. In S. Hoffmann, A. Sand, S. Arndt-Lappe, & L. M. Dillmann (Eds.), *Corpora and Lexis* (pp. 277–301). BRILL. https://doi.org/10.1163/9789004361133.
Biber, D. (1988). *Variation across speech and writing*. Cambridge University Press.
Biber, D. (2004). Historical patterns for the grammatical marking of stance: A cross-register comparison. *Journal of Historical Pragmatics, 5*(1), 107–136. https://doi.org/10.1075/jhp.5.1.06bib
Biber, D. (2006). Stance in spoken and written university registers. *Journal of English for Academic Purposes, 5*(2), 97–116. https://doi.org/10.1016/j.jeap.2006.05.001
Biber, D., Conrad, S., Reppen, R., Byrd, P., Helt, M., Clark, V., et al. (2004). Representing language use in the university: Analysis of the TOEFL 2000 spoken and written academic language corpus. *ETS TOEFL Monograph Series*, 25.
Biber, D., & Gray, B. (2013). Discourse characteristics of writing and speaking task types on the TOEFL IBT ® test: A Lexico-Grammatical analysis. *ETS Research Report Series, 2013*(1). https://doi.org/10.1002/j.2333-8504.2013.tb02311.x. i–128.
Biber, D., Gray, B., Staples, S., & Egbert, J. (2020). Investigating grammatical complexity in L2 English writing research: Linguistic description versus predictive measurement. *Journal of English for Academic Purposes, 100869*. https://doi.org/10.1016/j.jeap.2020.100869
Biber, D., Johansso, S., Leech, G., Conrad, S., & Finegan, E. (Eds.). (1999). *Longman grammar of spoken and written English (10. impression)*. Longman.
Biber, D., Johansson, S., Leech, G. N., Conrad, S., & Finegan, E. (2021). *Grammar of spoken and written English*. John Benjamins Publishing Company. https://doi.org/10.1075/z.232
Blanchard, D., Tetreault, J., Higgins, D., Cahill, A., & Chodorow, M. (2013). TOEFL11: A corpus of non-native English. *ETS Research Report Series, 2013*(2). https://doi.org/10.1002/j.2333-8504.2013.tb02331.x. i–15.
Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., …, & Amodei, D. (2020). Language models are few-shot learner. *arXiv:2005.14165*. http://arxiv.org/abs/2005.14165.
Carlson, L., Marcu, D., & Okurovsky, M. E. (2001). Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *, 2001*. *Proceedings of the Second SIGdial Workshop on Discourse and Dialogue*. SIGDIAL. https://aclanthology.org/W01-1605.
Chang, P., & Schleppegrell, M. (2011). Taking an effective authorial stance in academic writing: Making the linguistic resources explicit for L2 writers in the social sciences. *Journal of English for Academic Purposes, 10*(3), 140–151. https://doi.org/10.1016/j.jeap.2011.05.005
Chen, X., & Meurers, D. (2016). CTAP: A web-based tool supporting automatic complexity analysis. In D. Brunato, F. Dell'Orletta, G. Venturi, T. François, & P. Blache (Eds.), *Proceedings of the Workshop on Computational Linguistics for Linguistic Complexity (CL4LC)* (pp. 113–119). The COLING 2016 Organizing Committee. https://aclanthology.org/W16-4113.
Clark, K., Khandelwal, U., Levy, O., & Manning, C. D. (2019). What does BERT look At? An analysis of BERT's attention. *arXiv:1906.04341* [Cs] http://arxiv.org/abs/1906.04341.
Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement, 20*(1), 37–46. https://doi.org/10.1177/001316446002000104
Cotos, E., & Pendar, N. (2015). Discourse classification into rhetorical functions for AWE feedback. *CALICO Journal, 0*(0). https://doi.org/10.1558/cj.v33i1.27047
Crossley, S. A., Kyle, K., & Dascalu, M. (2019). The tool for the automatic analysis of cohesion 2.0: Integrating semantic similarity and text overlap. *Behavior Research Methods, 51*(1), 14–27. https://doi.org/10.3758/s13428-018-1142-4
Crosthwaite, P., & Jiang, K. (2017). Does EAP affect written L2 academic stance? A longitudinal learner corpus study. *System, 69*, 92–107. https://doi.org/10.1016/j.system.2017.06.010
Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*. https://doi.org/10.48550/arXiv.1810.04805
Dror, R., Peled-Cohen, L., Shlomov, S., & Reichart, R. (2020). *Statistical significance testing for natural language processing*. Morgan & Claypool.
Eguchi, M., & Kyle, K. (2023). Span identification of epistemic stance-taking in academic written English. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)* (pp. 429–442). https://aclanthology.org/2023.bea-1.35.
Ellis, R., & Barkhuizen, G. P. (2005). *Analysing learner language*. Oxford University Press.
Firth, J. R. (1957). A synopsis of linguistic theory, 1930-1955. In J. R. Firth (Ed.), *Studies in linguistic analysis*. Basil Blackwell.
Fuoli, M. (2018). A stepwise method for annotating appraisal. *Functions of Language, 25*(2), 229–258. https://doi.org/10.1075/fol.15016.fuo
Goldberg, A. E. (2003). Constructions: A new theoretical approach to language. *Trends in Cognitive Sciences, 7*(5), 219–224. https://doi.org/10.1016/S1364-6613(03)00080-9
Goldberg, Y. (2017). *Neural network methods for natural language processing*. Morgan & Claypool Publishers. https://doi.org/10.2200/S00762ED1V01Y201703HLT037
Graesser, A. C., McNamara, D. S., Louwerse, M. M., & Cai, Z. (2004). Coh-Metrix: Analysis of text on cohesion and language. *Behavior Research Methods, Instruments, & Computers, 36*(2), 193–202. https://doi.org/10.3758/BF03195564
Gu, W., Zheng, B., Chen, Y., Chen, T., & Van Durme, B. (2022). An empirical study on finding spans. *arXiv:2210.06824*. http://arxiv.org/abs/2210.06824.
Hagiwara, M. (2022). *Real-world natural language processing*. Manning Publications.
Halliday, M. A. K., & Matthiessen, C. M. I. M. (2014). *An introduction to functional grammar* (4th ed). Routledge.
Honnibal, M., Ines, M., Van Landeghem, S., & Boyd, A. (2020). spaCy: Industrial-strength natural language processing in python (Version 3.3) [Computer software]. https://spacy.io.
Hou, S., Zhang, S., & Fei, C. (2020). Rhetorical structure theory: A comprehensive review of theory, parsing methods and applications. *Expert Systems with Applications, 157*, Article 113421. https://doi.org/10.1016/j.eswa.2020.113421
Housen, A., Kuiken, F., & Vedder, I. (2012). *Dimensions of L2 performance and proficiency: Complexity, accuracy and fluency in SLA (Vol. 32)*. John Benjamins Publishing.
Huang, Y., Murakami, A., Alexopoulou, T., & Korhonen, A. (2018). Dependency parsing of learner English. *International Journal of Corpus Linguistics, 23*(1), 28–54. https://doi.org/10.1075/ijcl.16080.hua
Hyland, K. (2005). Metadiscourse: Exploring interaction in writing. *Continuum (Society for Social Work Administrators in Health Care)*.
Ishikawa, S. (2018). The ICNALE edited essays; a dataset for analysis of L2 English learner essays based on a new integrative viewpoint. *English Corpus Studies, 25*, 117–130.
Jurafsky, D., & Martin, J. H. (2009). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition* (2nd ed). Pearson Prentice Hall.
Kim, M., & Lu, X. (2024). Exploring the potential of using ChatGPT for rhetorical move-step analysis: The impact of prompt refinement, few-shot learning, and fine-tuning. *Journal of English for Academic Purposes, 71*, Article 101422. https://doi.org/10.1016/j.jeap.2024.101422
Kormos, J. (2011). Task complexity and linguistic and discourse features of narrative writing performance. *Journal of Second Language Writing, 20*(2), 148–161. https://doi.org/10.1016/j.jslw.2011.02.001
Kuiken, F., & Vedder, I. (2017). Functional adequacy in L2 writing: Towards a new rating scale. *Language Testing, 34*(3), 321–336. https://doi.org/10.1177/0265532216663991
Kyle, K. (2016). Measuring Syntactic Development in L2 Writing: Fine Grained Indices of Syntactic Complexity and Usage-Based Indices of Syntactic Sophistication.
Kyle, K., Crossley, S. A., & Berger, C. (2018). The tool for the automatic analysis of lexical sophistication (TAALES): Version 2.0. *Behavior Research Methods, 50*(3), 1030–1046. https://doi.org/10.3758/s13428-017-0924-4
Kyle, K., & Eguchi, M. (2021). Automatically assessing lexical sophistication using words, n-gram, and dependency bigram indices. In S. Granger (Ed.), *Perspectives on the second language phrasicon: The view from learner corpora*. Multilingual Matters.

Kyle, K., & Eguchi, M. (2024). Evaluating NLP models with written and spoken L2 samples. *Research Methods in Applied Linguistics, 3*(2), Article 100120. https://doi.org/10.1016/j.rmal.2024.100120

Kyle, K., & Sung, H. (2023). An Argument Structure Construction Treebank. In *The First International Workshop on Construction Grammars and NLP (CxGs+NLP, GURT/SyntaxFest 2023)* (pp. 51–62).

Lam, S. L., & Crosthwaite, P. (2018). APPRAISAL resources in L1 and L2 argumentative essays: A contrastive learner corpus-informed study of evaluative stance. *Journal of Corpora and Discourse Studies, 1*(1), 8. https://doi.org/10.18573/jcads.1

Lancaster, Z. (2014). Exploring valued patterns of stance in upper-level student writing in the disciplines. *Written Communication, 31*(1), 27–57. https://doi.org/10.1177/0741088313515170

Landis, J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics, 33*(1), 159–174. https://doi.org/10.2307/2529310

Lee, S. H. (2010). Attribution in high-and low-graded persuasive essays by tertiary students. *Functions of Language, 17*(2), 181–206. https://doi.org/10.1075/fol.17.2.02lee

Lee, S. H. (2017). Use of implicit intertextuality by undergraduate students: Focusing on Monogloss in argumentative essays. *Linguistics & the Human Sciences, 13*(1), 150–178. https://doi.org/10.1558/lhs.30651

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., et al. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv: Computation and Language*.

Lu, X. (2011). A corpus-based evaluation of syntactic complexity measures as indices of college-level ESL Writers' language development. *TESOL Q., 45*(1), 36–62. https://doi.org/10.5054/tq.2011.240859

Lu, X. (2012). The relationship of lexical richness to the quality of ESL learners' oral narratives. *Modern Language Journal, 96*(2), 190–208. https://doi.org/10.1111/j.1540-4781.2011.01232_1.x

Martin, J. R., & White, P. R. R. (2005). *The language of evaluation: Appraisal in English*. Palgrave Macmillan.

McHugh, M. L. (2012). Interrater reliability: The kappa statistic. *Biochemia Medica, 22*(3), 276–282. https://doi.org/10.11613/BM.2012.031

Meurers, D., & Dickinson, M. (2017). Evidence and interpretation in language learning research: Opportunities for collaboration with computational linguistics. *Language Learning*, (June), 66–95. https://doi.org/10.1111/lang.12233

Mizumoto, A., Shintani, N., Sasaki, M., & Teng, M. F. (2024). Testing the viability of ChatGPT as a companion in L2 writing accuracy assessment. *Research Methods in Applied Linguistics, 3*(2), Article 100116. https://doi.org/10.1016/j.rmal.2024.100116

Norris, J.M., & Ortega, L. (2003). Defining and measuring SLA. In C. Doughty & M. H. Long (Eds.), The handbook of second language acquisition.

Norris, J. M., & Ortega, L. (2009). Towards an organic approach to investigating CAF in instructed SLA: The case of complexity. *Applied Linguistics, 30*(4), 555–578. https://doi.org/10.1093/applin/amp044

Ortega, L. (2009). *Understanding second language acquisition. routledge*. Taylor & Francis Group.

Pallotti, G. (2009). CAF: Defining, refining and differentiating constructs. *Applied Linguistics, 30*(4), 590–601. https://doi.org/10.1093/applin/amp045

Palmer, F. R. (2001). *Mood and modality* (2nd ed). Cambridge University Press.

Papay, S., Klinger, R., & Padó, S. (2020). Dissecting span identification tasks with performance prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 4881–4895). https://doi.org/10.18653/v1/2020.emnlp-main.396

Paquot, M. (2019). The phraseological dimension in interlanguage complexity research. *Second Language Research, 35*(1), 121–145. https://doi.org/10.1177/0267658317694221

Park, K., & Lu, X. (2015). Automatic analysis of thematic structure in written English. *International Journal of Corpus Linguistics, 20*(1), 81–101. https://doi.org/10.1075/ijcl.20.1.04par

Qin, J., & Liu, D. (2024). Introducing/Testing New SFL-inspired communication/content/function-focused measures for assessing L2 narrative task performance. *Applied Linguistics*, amae030. https://doi.org/10.1093/applin/amae030

Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. https://www.mikecaptain.com/resources/pdf/GPT-1.pdf.

Read, J., & Carroll, J. (2012). Annotating expressions of Appraisal in English. *Language Resources and Evaluation, 46*(3), 421–447. https://doi.org/10.1007/s10579-010-9135-7

Römer, U., & Swales, J. M. (2010). The Michigan corpus of upper-level student papers (MICUSP). *Journal of English for Academic Purposes, 9*(3), 249.

Shatz, I. (2020). Refining and modifying the EFCAMDAT: Lessons from creating a new corpus from an existing large-scale English learner language database. *International Journal of Learner Corpus Research, 6*(2), 220–236. https://doi.org/10.1075/ijlcr.20009.sha

Skehan, P. (2009). Lexical performance by native and non-native speakers on language-learning tasks. *Vocabulary Studies in First and Second Language Acquisition: The Interface between Theory and Application*, 107–124. https://doi.org/10.1057/9780230242258

Sung, H., & Kyle, K. (2024). Annotation scheme for english argument structure constructions treebank. 12–18.

Swales, J. M. (1990). *Genre analysis: English in academic and research settings*. Cambridge University Press.

Thampi, A. (2022). *Interpretable AI*. Manning Publications.

Tunstall, L., Werra, L.von, & Wolf, T. (2022). *Natural language processing with transformers: Building language applications with hugging face* (1st edition). O'Reilly Media.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems, 30*. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Wang, S., Sun, X., Li, X., Ouyang, R., Wu, F., Zhang, T., Li, J., & Wang, G. (2023). GPT-NER: Named entity recognition via large language models. *arXiv:2304.10428*. https://doi.org/10.48550/arXiv.2304.10428

Wang, X., & Wang, Y. (2022). Sentence-level resampling for named entity recognition. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 2151–2165). https://doi.org/10.18653/v1/2022.naacl-main.156

White, P. R. R. (2003). Beyond modality and hedging: A dialogic view of the language of intersubjective stance. *Text - Interdisciplinary Journal for the Study of Discourse, 23*(2). https://doi.org/10.1515/text.2003.011

Wolfe-Quintero, K., Inagaki, S., & Kim, H.-Y. (1998). *Second language development in writing: Measures of fluency, accuracy, & complexity*. University of Hawaii Press.

Wu, S. M. (2007). The use of engagement resources in high- and low-rated undergraduate geography essays. *Journal of English for Academic Purposes, 6*(3), 254–271. https://doi.org/10.1016/j.jeap.2007.09.006

Xu, Y. (2020). *Second language writing complexity in academic legal discourse: Development and assessment under a curricular lens*. Georgetown University [PhD Dissertation].

Yannakoudakis, H., Briscoe, T., & Medlock, B. (2011). A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies* (pp. 180–189). https://aclanthology.org/P11-1019.

Yimam, S. M., Gurevych, I., Eckart de Castilho, R., & Biemann, C. (2013). WebAnno: A flexible, web-based and visually supported system for distributed annotations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (pp. 1–6). https://www.aclweb.org/anthology/P13-4001.

Yoon, H.-J. (2017). Textual voice elements and voice strength in EFL argumentative writing. *Assessing Writing, 32*, 72–84. https://doi.org/10.1016/j.asw.2017.02.002