Assisting Undergraduate Students in Writing Spanish Methodology Sections

Samuel González-López
Technological University of Nogales
Nogales, Sonora, México
sgonzalez@utnogales.edu.mx

Steven Bethard
University of Arizona
Tucson, Arizona, USA
bethard@email.arizona.edu

Aurelio López-López
National Institute of Astrophysics, Optics and Electronics
Tonantzintla, Puebla, México
allopez@inaoep.mx

Abstract

In undergraduate theses, a good methodology section should describe the series of steps that were followed in performing the research. To assist students in this task, we develop machine-learning models and an app that uses them to provide feedback while students write. We construct an annotated corpus that identifies sentences representing methodological steps and labels when a methodology contains a logical sequence of such steps. We train machine-learning models based on language modeling and lexical features that can identify sentences representing methodological steps with 0.939 f-measure, and identify methodology sections containing a logical sequence of steps with an accuracy of 87%. We incorporate these models into a Microsoft Office Add-in, and show that students who improved their methodologies according to the model feedback received better grades on their methodologies.

1 Introduction

In the Mexican higher education system, most undergraduate students write a thesis (*tesis de licenciatura*) before graduation. The academic advisor and the student are typically both involved. Throughout the process, the advisor spends time reviewing the draft that the student is building and gradually offering suggestions. This process becomes a cycle until the document meets established standards and/or institutional guidelines. This cycle is often slow due to the required changes in the structure of the thesis. One of the key components of such a thesis is a methodology section, which contains the steps and procedures used to develop the research. A methodology is supposed to provide a step-by-step explanation of the aspects necessary to understand and replicate the research including the techniques and procedures employed, the type of research, the population studied, the data sample, the collection instruments, the data selection process, the validation instrument, and the statistical analysis process (Allen, 1976).

Natural language processing techniques have the potential to assist students in writing such methodologies, as several aspects of good methodologies are visible from lexical and orthographic features of the text. A good methodology should have phrases or sentences that represent a series of steps, which may be written in a numbered list or in prose with sequential connectives like *next*. Steps in a methodology section should have a predicate that represents the action of that step, like *analyze* or *design*. And the list of steps should be in a logical order, e.g., an *explore* step should typically appear before (not after) an *implement* step. Good methodology sections should of course have much more beyond these simple features, but any methodology section that is missing these basic components is clearly in need of revision.

We thus focus on designing machine-learning models to detect and evaluate the quality of such steps in a Spanish-language student-written methodology section, and on incorporating such models into an interactive application that gives students feedback on their writing. Our contributions are the following:

- We annotate a small corpus of methodology sections drawn from Spanish information technology theses for the presence of steps and their logical order.
- We design a model to detect sentences that represent methodological steps, incorporating language model and verb taxonomy features, achieving 0.939 f-measure.

Proceedings of the 15th Workshop on Innovative Use of NLP for Building Educational Applications, pages 115–123
July 10, 2020. ©2020 Association for Computational Linguistics

- We design a model to identify when a methodology has a logical sequence of steps, incorporating language model and content word features, achieving an accuracy of 87%.
- We incorporate the models into an Add-In for Microsoft Word, and measure how the application's feedback improves student writing.

2 Background

There is a long history of natural language processing research on interactive systems that assist student writing. Essay scoring has been a popular topic, with techniques ranging from syntactic and discourse analysis (Burstein and Chodorow, 1999), to list-wise learning-to-rank (Chen and He, 2013), to recurrent neural networks (Taghipour and Ng, 2016). Yet the goal of such work is very different from ours, as we aim not to assign an overall score, but rather to provide detailed feedback on aspects of a good methodology that are present or absent from the draft.

Intelligent tutoring systems have been developed for a wide range of topics, including mechanical systems (Di Eugenio et al., 2002), qualitative physics (Litman and Silliman, 2004), learning a new language (Wang and Seneff, 2007), and introductory computer science (Fossati, 2008). As we focus on assisting students in writing thesis methodology sections, the most relevant prior work focuses on analysis of essays. ETS Criterion (Attali, 2004) uses features like n-gram frequency and syntactic analysis to provide feedback for grammatical errors, discourse structure, and undesirable stylistic features. The SAT system (Andersen et al., 2013) combines lexical and grammatical properties with a perceptron learner to provide detailed sentence-by-sentence feedback about possible lexical and grammatical errors. Revision Assistant (Woods et al., 2017) uses logistic regression over lexical and grammatical features to provide feedback on how individual sentences influence rubric-specific formative scores. All of these systems aim at general types of feedback, not the specific feedback needed for methodology sections.

Other related work touches on issues of logical organization. Barzilay and Lapata (2008) propose training sentence ordering models to differentiate between the original order of a well-written text and a permuted sentence order. Cui et al. (2018) continue in this paradigm, training an encoder-decoder network to read a series of sentences and reorder them for better coherence. Our goal is not to reorder a student's sentences, but to provide more detailed feedback on whether the right structures (e.g., steps) are present in the methodology. More relevant work is Persing et al. (2010), which combines lexical heuristics with sequence alignment models to score the organization of an essay. However, they provide only an overall score, and do not integrate this into any intelligent tutoring system.

A final major difference between our work and prior work is that all the work above focused on the English language, while we provide feedback for Spanish-language theses.

3 Data

A collection was created using the ColTyPi[1] site. This site includes Spanish-language theses within the Information Technologies subject area. The graduate level is composed of Doctoral and Master theses. The Undergraduate level is composed of Bachelor and Advanced College-level Technician (TSU) theses. All theses and research proposals in the collection have been reviewed at some point by a review committee.

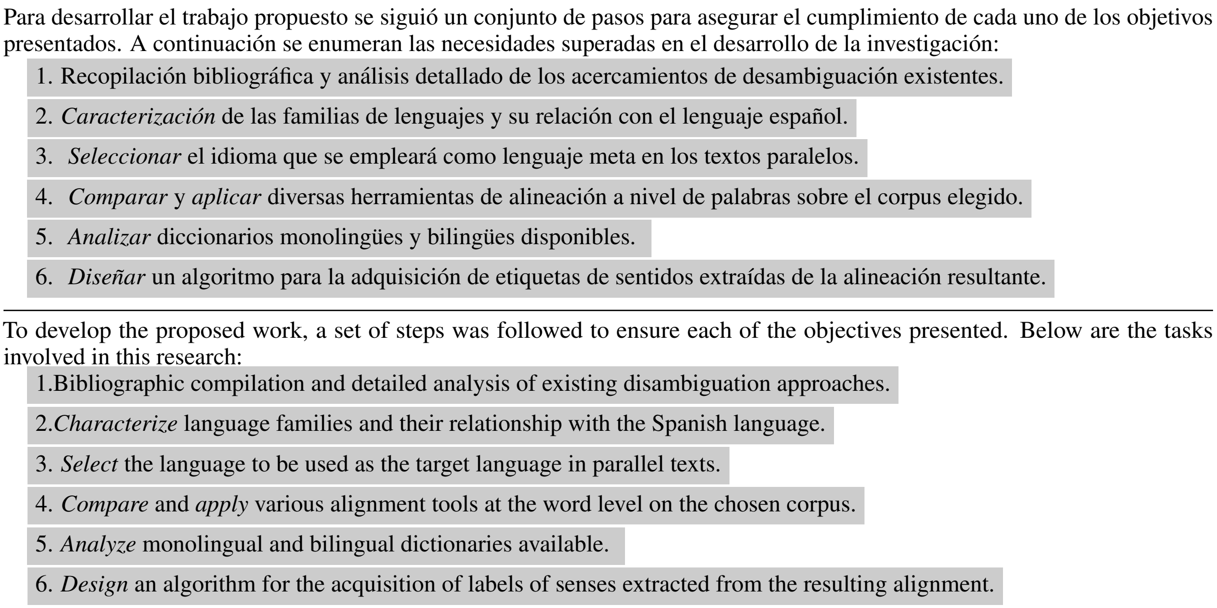
3.1 Guidelines

A four-page guide was provided to the annotators with the instructions for labeling and a brief description of the elements to identify. Annotators marked each sentence (or text segment) that represented a step in a series of steps. For each step, annotators marked the main predicate (typically a verb). Finally, annotators judged whether or not the steps of the methodology represented a logical sequence.The guide included three examples for the annotators, the first one detailed a methodology that accomplished a series of steps and a logical sequence, the second example only met a series of steps, and the third example didn't show any feature. The annotators did not have access to the academic corresponding to each methodology. Figure 1 shows an annotated example.

3.2 Annotation

From ColTyPi, 160 methodologies were downloaded, 40 at the PhD level, 60 at the Master level, 40 at the Bachelor level, and 20 at the TSU level. Two professors in the computer area with experience in reviewing graduate and undergraduate

[1] We used, `http://coltypi.org/`



Para desarrollar el trabajo propuesto se siguió un conjunto de pasos para asegurar el cumplimiento de cada uno de los objetivos presentados. A continuación se enumeran las necesidades superadas en el desarrollo de la investigación:

1. Recopilación bibliográfica y análisis detallado de los acercamientos de desambiguación existentes.
2. *Caracterización* de las familias de lenguajes y su relación con el lenguaje español.
3. *Seleccionar* el idioma que se empleará como lenguaje meta en los textos paralelos.
4. *Comparar* y *aplicar* diversas herramientas de alineación a nivel de palabras sobre el corpus elegido.
5. *Analizar* diccionarios monolingües y bilingües disponibles.
6. *Diseñar* un algoritmo para la adquisición de etiquetas de sentidos extraídas de la alineación resultante.

To develop the proposed work, a set of steps was followed to ensure each of the objectives presented. Below are the tasks involved in this research:

1.Bibliographic compilation and detailed analysis of existing disambiguation approaches.

2.*Characterize* language families and their relationship with the Spanish language.

3. *Select* the language to be used as the target language in parallel texts.

4. *Compare* and *apply* various alignment tools at the word level on the chosen corpus.

5. *Analyze* monolingual and bilingual dictionaries available.

6. *Design* an algorithm for the acquisition of labels of senses extracted from the resulting alignment.

Figure 1: Part of a Spanish methodology tagged by the annotators (Spanish original above, English translation below). The series of steps is shaded in gray, the verbs identified are in italics, and the annotators marked this methodology as "Yes" for the presence of logical sequence.

student theses, were recruited as annotators. Both annotators tagged 160 methodology sections, and inter-annotator agreement was measured. For the two information extraction tasks, identifying steps and identifying predicates, inter-annotator agreement was measured with F-score following Hripcsak and Rothschild (2005). For logical sequence, which is a binary per-methodology judgment, inter-annotator agreement was measured with Cohen's Kappa (Landis and Koch, 1977). The annotators achieved 0.90 F-score on identifying steps, 0.89 F-score on identifying predicates, and 0.46 Kappa (moderate agreement) on judging logical sequence. Identifying the logical sequence was a complicated task for the annotators since the objective was that a whole methodology evidenced a logical sequence concerning the verbs used. For instance, in the first steps of the methodology, the student should use verbs like "identify" or "explore" and verbs like "implement" or "install" at the end of the methodology.

The annotated data was divided up for experiments. Only annotations that both annotators agreed on were considered. For the methodological step extraction task, we selected 300 sentences annotated as representing a step, and 100 sentences annotated as not representing a step, with the sentences selected to cover both graduate and undergraduate levels. For the logical sequence detection task, we selected 50 complete methodologies annotated as having a logical sequence and 50 annotated as not having a logical sequence.

4 Model: step identification

The model for identifying which sentences represent steps (StepID) is a logistic regression[2] that takes a sentence as input, and predicts whether that sentence is a methodology step or not. The model considers the five types of features described in the following sections.

4.1 Language model features

To measure how well the words in a Methodology match the typical sequence of words in a good Methodology, we turn to language modeling techniques. We expected to capture facts like that the presence of verbs "Select', "Analyze" or "Compare" at the beginning of sentences is probably describing a series of steps. We preprocessed all sentences by extracting lemmas using FreeLing.[3] Afterwards, two language models were built, the first (TM) with tokens (words, numbers, punctuation marks) and the second (GM) with grammatical classes. These language models were built only on the sentences labeled as positive, i.e., on sentences that should be examples of good token/grammatical

[2] We used the implementation in Weka 3.6.13, https://www.cs.waikato.ac.nz/ml/weka/

[3] FreeLing4.1,http://nlp.lsi.upc.edu/freeling/

class sequences. We used the SRILM[4] toolkit with 4-grams and Kneser-Ney smoothing.[5] To generate these features for the 300 positive sentences on which the language models were trained, we used 10-fold cross-validation, so as not to overestimate the language model probabilities. The 100 negative sentences were also processed separately, again with a 10-fold cross-validation.Perplexity values from the language models were used as features. This component contributed 2 features to the StepID classifier.

4.2 Sentence location features

A methodology can begin immediately with sequence of steps, or there may be a brief introduction before the steps appear. Thus, location within the methodology may be a predictive feature. We identified whether the sentence under consideration is in the first third, second third, or final third of the methodology. This component contributed 3 features to the StepID classifier.

4.3 Verb taxonomy features

This component captures the type of the verbs used in the series of steps. We use a taxonomy based on the cyclical nature of engineering education (CNEE; Fernandez-Sanchez et al., 2012), structured in four successive levels. Categories of verbs include Knowledge and Comprehension, Application and Analysis, System Design, Engineering Creation. In addition, we added a category to identify verbs related to the writing process, as part of the steps to conclude the thesis.

We considered three ways of identifying such verb categories in sentences.

CNEE+Stem Each verb in the sentence is stemmed, and compared against the 54 verbs of the CNEE taxonomy.

CNEE+FastText The 54 verbs in the CNEE taxonomy are expanded to 540 verbs by taking the 10 most similar words according to pretrained word vectors from FastText (Bojanowski et al., 2016)[6]. Each verb in the sentence is compared against these 540 verbs.

CNEE+Manual An expert annotator manually labeled each verb with an appropriate one of the five categories from the CNEE taxonomy.

For CNEE+Stem and CNEE+FastText, only the first verb category found is included as a feature[7]. This component contributed 5 features to the StepID classifier.

4.4 Sequencing element features

The online writing lab at Purdue University[8] identifies a category of words designed “to show sequence” that includes words like *first*, *second*, *next*, *then*, *after*. We coupled the words from this category with a simple pattern to identify bullet points or numbered items to produce a rule that identifies whether such sequencing elements are present in the text. This component contributed 1 feature to the StepID classifier.

5 Model: logical sequence detection

The model for detecting logical sequence (LogicSeq) is a multilayer perceptron, with a single hidden layer of size two plus the number of features (Weka’s `a` layer specifier), that takes an entire methodology as input, and predicts whether it contains a logical sequence of steps or not. The model considers the features described in the following section.

5.1 Language model features

We again incorporate language models to measure how sequences of terms are used in well-written methodologies. This component includes the same GM and TM features as Section 4.1, except trained on the 100 positive and negative methodologies, rather than on individual sentences. We also include a third language model that considers only the nouns and verbs (NV) of the sentences of the methodology. Each token is followed by its part of speech in the language model input. The goal is to focus on just the words most likely to express methodological steps – *characterize*, *select*, *compare*, *analyze*, *design*, etc. – without restricting the analysis to a specific lexicon of words.

We considered bigrams and/or 4-grams for the GM, TM, and NV features. This component contributed either 3 features to the LogicSeq classifier, or 6 features when both bigrams and 4-grams were used.

[4] SRILM 1.7.3, http://www.speech.sri.com/projects/srilm/

[5] In preliminary experiments, we also tried using the TheanoLM toolkit, but performance was lower than SRILM.

[6] https://fasttext.cc/

[7] In preliminary experiments, we also tried using all verb categories, but this did not improve performance.

[8] https://owl.purdue.edu/

TM	GM	Loc	CNEE+Stem	CNEE+FastText	CNEE+Manual	Precision	Recall	F
✓						0.808	0.953	0.875
	✓					0.816	0.947	0.877
✓	✓					0.834	0.953	0.890
✓	✓	✓				0.843	0.947	0.892
✓	✓	✓	✓			0.901	0.937	0.918
✓	✓	✓		✓		0.912	0.967	**0.939**
✓	✓	✓			✓	0.949	0.983	0.966

Table 1: 10-fold cross-validation performance on the “is this a methodological step” classification task.

GM bigram	TM bigram	NV bigram	GM 4-gram	TM 4-gram	NV 4-gram	Accuracy
✓						76 %
	✓					72 %
		✓				63 %
✓	✓	✓				77 %
			✓	✓	✓	79 %
✓	✓	✓	✓	✓	✓	**87 %**

Table 2: 10-fold cross-validation performance on the “is there a logical sequence ” classification task.

6 StepID and LogicSeq results

Both classifiers were evaluated using 10-fold cross-validation on their respective parts of our annotated corpus.

Table 1 shows the performance of the step identification model in terms of precision, recall, and F-score for detecting steps. Including all proposed features proposed yields 0.918 when stemming is used to find verbs and 0.939 when FastText is used instead of stemming. Using the human-annotated verb features yields 0.966, suggesting that performance could be further improved with a better lexicon mapping technique.

Table 2 shows the performance of the logical sequence detection model. The best model used both bigrams and 4-grams of all three language-model features, and achieved an accuracy of 87%.

We thus find that despite our modest-sized data sets, accurate models based on language-model features can be trained to detect methodological steps in a thesis and identify whether those steps appear in a logical order. In the next section, we move from the intrinsic evaluation of our models on the annotated dataset to an extrinsic evaluation in a user study.

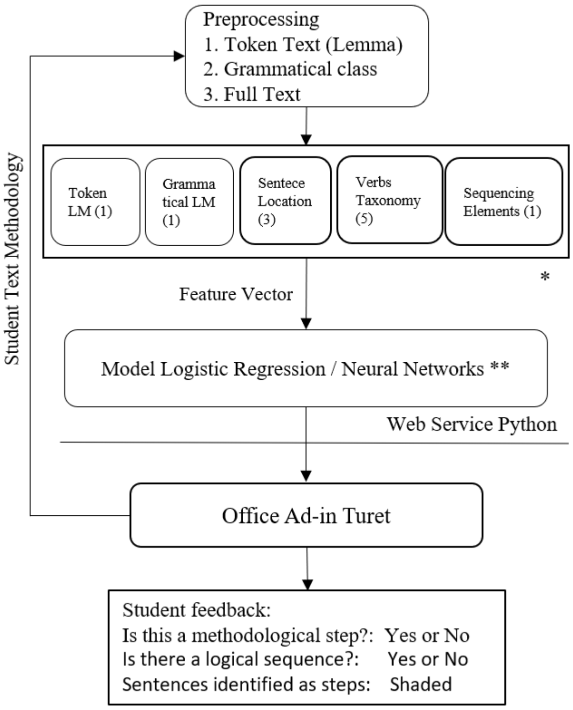


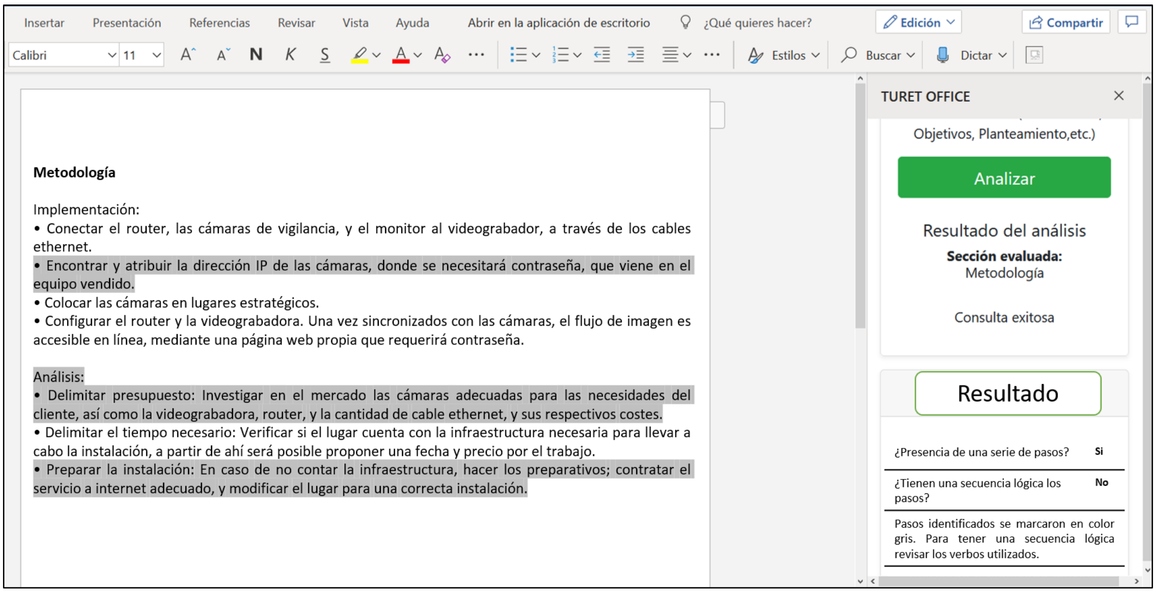
Figure 2: System architecture for TURET. *StepID used all features; LogicSeq used only LM features. **StepID used logistic regression; LogicSeq used neural networks.

7 Pilot test

We designed and performed a pilot test to assess the impact of using an application focused on the two models created, StepID and LogicSeq. The goal is to evaluate these models in an environment where students interact with the models while writing. Our main research question is: What elements incorporated in the developed methods will have a positive impact on the student’s final document?

7.1 User interface

We first developed an Office add-in that could apply the StepID and LogicSeq models to a document while students were writing it. We chose to implement the app as an Office add-in as it allowed students to work in a writing environment they were already very familiar with: Microsoft Word. The software developed, Tutor Revisor de Tesis



Metodología

Implementación:
• Conectar el router, las cámaras de vigilancia, y el monitor al videograbador, a través de los cables ethernet.
• Encontrar y atribuir la dirección IP de las cámaras, donde se necesitará contraseña, que viene en el equipo vendido.
• Colocar las cámaras en lugares estratégicos.
• Configurar el router y la videograbadora. Una vez sincronizados con las cámaras, el flujo de imagen es accesible en línea, mediante una página web propia que requerirá contraseña.

Análisis:
• Delimitar presupuesto: Investigar en el mercado las cámaras adecuadas para las necesidades del cliente, así como la videograbadora, router, y la cantidad de cable ethernet, y sus respectivos costes.
• Delimitar el tiempo necesario: Verificar si el lugar cuenta con la infraestructura necesaria para llevar a cabo la instalación, a partir de ahí será posible proponer una fecha y precio por el trabajo.
• Preparar la instalación: En caso de no contar la infraestructura, hacer los preparativos; contratar el servicio a internet adecuado, y modificar el lugar para una correcta instalación.

Objetivos, Planteamiento,etc.)

Resultado del análisis

Sección evaluada:
Metodología

Consulta exitosa

¿Presencia de una serie de pasos?	Si
¿Tienen una secuencia lógica los pasos?	No

Pasos identificados se marcaron en color gris. Para tener una secuencia lógica revisar los verbos utilizados.

Figure 3: Application interface embedded in Office Ad-in

(TURET), was embedded in the Microsoft Word processor through a component developed in the Azure platform for Office Add-ins. As part of this development, we had to re-implement the StepID and LogicSeq algorithms using Scikit-learn, but reused the same language model features created with the SRILM toolkit. Figure 2 shows the architecture of the system. In the first stage the preprocessing was done sentence by sentence to compute eleven features established in the StepId method. For the LogicSeq method the entire methodology was processed to extract six features.

Figure 3 shows an example methodology open in Microsoft Word with TURET enabled. The methodology written by the student is shown on the left side. After clicking, sentences that are identified as being part of a series of steps are marked, and the student is also sent binary feedback indicating whether the methodology shows a series of steps and/or a logical sequence. Notice that the methodology shows seven steps, but the method only detects 3 of them as valid. This is most likely because words like *implementation* and *connect* are not generally appropriate at the beginning of a methodology. Thus, this example shows an absence of a logical sequence. The system correctly predicts this, as shown in through the *No* in the feedback frame.

7.2 Experimental design

The pilot test was conducted with two groups of 20 (for a total of 40) undergraduate computer science students. Each student received an introduction explaining how to use the TURET application. Then the student was provided with a problem statement related to a computer science project and was asked to write a methodology that provides a solution. Students were encouraged to try to achieve positive feedback from the system on two aspects: that the methodology had a logical sequence and that there was evidence of a series of steps. Students had access to the application for 1 month and were expected to use TURET at least twice (i.e., on a first draft and a final draft) but could freely use the application more frequently if desired.

We also included a control group of 20 undergraduate computer science students who did not use TURET, but still used Microsoft Word to write a methodology in response to the same problem statements.

To validate the quality of the documents generated by both the TURET students and the control students, a teacher experienced in grading undergraduate theses evaluated both the first and the final draft. Each methodology received a rating on a scale of 1 to 10, where 10 is the best. The teacher was not informed about the use of the TURET application; they graded the methodologies as they would. Of the total number of students who started the pilot test, only 35 completed the entire process.

7.3 Statistical analysis

A multiple regression analysis was made on the results obtained from the evaluation of the method-

Factor	Coefficients	P-values
Intercept	7.5552	0.0001
N-Steps	-0.1421	0.1263
Steps?	0.2652	0.6946
Logical Sequence?	1.2237	0.0096

Table 3: Coefficients of different final draft factors when predicting the final grade of a student.

Items	Coefficients	P-values
Intercept	0.7768	0.0231
N-Steps	-0.3066	0.0097
Steps?	1.0103	0.0342
Logical Sequence?	0.6342	0.1270

Table 4: Coefficients of different (Final - Draft) factors when predicting the change in grade between Draft and Final (i.e., the Final - Draft grade).

ologies, with the teacher's grade of the final draft as the dependent variable and the following factors measured on their final draft:

N-Steps A non-negative integer representing the number of sentences of each methodology that the StepId model recognized as methodological steps.

Steps? A binary value, with a value of 1 when the StepId model recognized at least one sentence as a methodological step, or a value of 0 if there was no such sentence.

Logical Sequence? A binary value, with a value of 1 when the LogicSeq model recognized the methodology as having a logical sequence, and a value of 0 otherwise.

Table 3 shows that when predicting the grade assigned to a student's final draft, the LogicSeq model's prediction is a statistically significant predictor: drafts judged to have a logical sequence were on average score 1.2237 higher than the other drafts.

We also explored a multiple regression designed to test how much changes in a student's writing predicted changes in their grade. Instead of considering only the final draft, as above, we consider the difference between the initial and the final for all factors as well as the dependent variable. We thus re-define the factors as follows.

N-Steps An integer representing the increase in number of sentences recognized as methodological steps by the StepId model when moving from the draft to the final document.

Steps? An integer, with a value of 1 when the StepId model found no steps in the draft but at least one in the final, a value of 0 when the number of steps identified by StepId was unchanged between draft and final, and a value of -1 when the StepId model found at least one step in the draft but none in the final.

Logical Sequence? An integer, with a value of 1 when the LogicSeq model found no logical sequence in the draft but found one in the final, a value of 0 when there was no change in the prediction of the LogicSeq model between draft and final, and a value of -1 when the LogicSeq model found a logical sequence in the draft but none in the final.

Table 4 shows that when predicting how much a student's grade will improve from draft to final, the change in the number of steps identified by StepId is a statistically significant predictor. Students that went from having no steps to having one or more steps on average scored 1.0103 better than students with no change. Interestingly, having many steps was not necessarily a good thing: for each additional step, students on average lost 0.3066 from their score. This suggests that students who added too many more steps to their drafts were penalized for doing so.

Finally, we compared the TURET group of students against the control group of students. On the 20 problem statements that were common to the TURET and control groups, the TURET students on average scored 7.85, while the control students scored 6.8. The difference is significant ($p = .041139$) according to a t-test for two independent means (two tailed).

7.4 Satisfaction survey

To assess the opinion of the experimental group on using the TURET Office Add-in, a satisfaction survey based on the Technology Acceptance Model (Davis et al., 1989) was conducted. Students were asked about the usefulness, ease of use, adaptability and, their intention to use the system. For example, the "usefulness" questions were: Does the system improve your methodology? Did the system improve the performance of your learning? In general, do you think that the system was an advantage for your learning to write arguments? As another example, the "ease of use" questions were: Was learning to use the system easy for you? Was the process

Measure	Score
Usefulness	4.54
Ease of use	4.85
Adaptability	4.67
Intention to use	4.50

Table 5: Satisfaction survey results TAM

of using the system clear and understandable?. In general, do you think the system was easy to use?

Student answers were based on a five-point Likert scale ranging from 1 (“Strongly disagree”) to 5 (“Strongly agree”), and the scores across each category of question were averaged. Table 5 shows that students rated the application above 4 points (“Agree”) for all aspects. The highest score was 4.85 on ease of use, which we attribute to the use of a Microsoft Word Add-in, which takes advantage of students’ already existing familiarity with Microsoft Word.

We also collected free-form comments from the students. Their biggest complaint was that TURET works only in the online version of Microsoft Office (since it must communicate with a server), and they would have liked to use it in offline mode.

8 Discussion

We have demonstrated that with a small amount of training data, several carefully engineered features, and standard supervised classification algorithms, we can construct models that can reliably (0.939 F) detect the presence of steps in student-written Spanish methodology sections, and reliably (87% accuracy) determine whether those steps are presented in a logical order. We have also shown that incorporating these models into an Office Add-in for Microsoft Word resulted in a system that students found useful and easy to use, and that the detections of the models were predictive of teacher-assigned essay grades.

There are some limitations to our study. First, because of the success of our simple models, we did not investigate more complex recent models like BERT (Devlin et al., 2019). Such models might yield improved predictive performance but at a significant additional computational cost. Second, the amount of data that we annotated was small, as it required a high level of expertise in the reviewing of Spanish-language methodology sections. (We relied on Spanish-speaking professors in computer science.) It would be good to expand the size of the dataset, but we take the high levels of performance of the models, and the fact that they make useful predictions on the unseen student-generated methodologies of the pilot test, as an indication that the dataset is already useful in its current size. Finally, the pilot study was a controlled experiment, where specific problem statements were given as prompts. It would be interesting to measure the utility of the application for students writing their own theses.

In the future, we would like to explore integrating other types of writing feedback into the TURET Office Add-in, since students found its feedback about methodology steps both intuitive and helpful. Though we focused on methodology sections in this article, our vision is a set of models that can provide useful feedback for all sections of a Spanish-language student thesis.

References

George R. Allen. 1976. *The Graduate Students’ Guide to Theses and Dissertations: A Practical Manual for Writing and Research*, volume 1. Jossey-Bass Inc., Publishers, 615 Montgomery Street, San Francisco.

Øistein E. Andersen, Helen Yannakoudakis, Fiona Barker, and Tim Parish. 2013. Developing and testing a self-assessment and tutoring system. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 32–41, Atlanta, Georgia. Association for Computational Linguistics.

Yigal Attali. 2004. Exploring the feedback and revision features of criterion. *Journal of Second Language Writing*, 14:191–205.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Jill Burstein and Martin Chodorow. 1999. Automated essay scoring for nonnative English speakers. In *Computer Mediated Language Assessment and Evaluation in Natural Language Processing*.

Hongbo Chen and Ben He. 2013. Automated essay scoring by maximizing human-machine agreement. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1741–1752, Seattle, Washington, USA. Association for Computational Linguistics.

Baiyun Cui, Yingming Li, Ming Chen, and Zhongfei Zhang. 2018. Deep attentive sentence ordering network. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4340–4349, Brussels, Belgium. Association for Computational Linguistics.

Fred D. Davis, Richard P. Bagozzi, and Paul R. Warshaw. 1989. User acceptance of computer technology: A comparison of two theoretical models. *Management Science*, 35(8):982–1003.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Barbara Di Eugenio, Michael Glass, and Michael Trolio. 2002. The DIAG experiments: Natural language generation for intelligent tutoring systems. In *Proceedings of the International Natural Language Generation Conference*, pages 120–127, Harriman, New York, USA. Association for Computational Linguistics.

Pilar Fernandez-Sanchez, Angel Salaverría, and Enrique Mandado. 2012. Taxonomía de los niveles del aprendizaje de la ingeniería y su implementación mediante herramientas informáticas. In *X Congreso de Tecnologías Aplicadas en la Enseñanza de la Electrónica*, pages 522–527.

Davide Fossati. 2008. The role of positive feedback in Intelligent Tutoring Systems. In *Proceedings of the ACL-08: HLT Student Research Workshop*, pages 31–36, Columbus, Ohio. Association for Computational Linguistics.

George Hripcsak and Adam S. Rothschild. 2005. Agreement, the f-measure, and reliability in information retrieval. *Journal of the American Medical Informatics Association*, 12(3):296–298.

J. Richard Landis and Gary G. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Diane J. Litman and Scott Silliman. 2004. ITSPOKE: An intelligent tutoring spoken dialogue system. In *Demonstration Papers at HLT-NAACL 2004*, pages 5–8, Boston, Massachusetts, USA. Association for Computational Linguistics.

Isaac Persing, Alan Davis, and Vincent Ng. 2010. Modeling organization in student essays. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 229–239, Cambridge, MA. Association for Computational Linguistics.

Kaveh Taghipour and Hwee Tou Ng. 2016. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1891, Austin, Texas. Association for Computational Linguistics.

Chao Wang and Stephanie Seneff. 2007. Automatic assessment of student translations for foreign language tutoring. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 468–475, Rochester, New York. Association for Computational Linguistics.

Bronwyn Woods, David Adamson, Shayne Miel, and Elijah Mayfield. 2017. Formative essay feedback using predictive scoring models. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 2071–2080, New York, NY, USA. Association for Computing Machinery.