The Popstar, the Poet, and the Grinch: Relating Artificial Intelligence to the Computational Thinking Framework with Block-based Coding

Jessica Van Brummelen[1*], Judy Hanwen Shen[2], Evan W. Patton[3]
[123]Massachusetts Institute of Technology
jess@csail.mit.edu, judyshen@mit.edu, ewpatton@mit.edu

ABSTRACT

As the world becomes increasingly saturated with artificial intelligence (AI) technology, computational thinking (CT) frameworks must be updated to incorporate AI concepts. In this paper, we propose five AI-related computation concepts, practice, and perspective: classification, prediction, generation, training/validating/testing, and evaluation. We propose adding them to a widely-used CT framework and present an MIT App Inventor extension that explores this framework through project-based learning.

KEYWORDS

artificial intelligence, conversational artificial intelligence, machine learning, computational thinking, K-12 education

1. INTRODUCTION

There are many who would like to understand and use artificial intelligence (AI) models but lack the tools and knowledge to do so. We propose adding AI-related concepts to Brennan and Resnick's (2012) CT framework, as well as present block-based coding tools to democratize AI education and programming. Our proposed tools were developed for MIT App Inventor, an open-source platform that enables anyone to develop mobile apps using block-based coding with eight million registered users from primary-school aged students to working-class adults (MIT App Inventor, 2017).

Malyn-Smith et. al. (2018) developed a CT framework from a disciplinary perspective which includes machine learning as an element. To this end, we propose computational concepts, practice, and perspective that more effectively capture the skills and competencies necessary to understand AI. Using conversational AI, AI-related components, including classification, prediction, generation, training/validating/testing, and evaluation, are explained. We present an MIT App Inventor extension to enable students to learn the proposed AI CT components.

2. EXTENDING CT WITH AI

Artificial intelligence can be understood within a symbolic-rule/machine-learning paradigm. In symbolic rule-based AI, collections of *if-then* statements or other rules determine how AI agents behave (Winston & Shellard, 1990). In machine learning-based AI, machines determine how to behave through extracting patterns. Both methods have shortcomings, such as the difficulty of programming an exhaustive list of rules for rule-based AI, and the limited interpretability of machine learning models.

Within the symbolic-rule/machine-learning paradigm, designers use AI to classify, predict, and generate information. For example, an autonomous vehicle may perceive objects on the road and *classify* them as "pedestrians" or "motor vehicles"; *predict* objects' motion; and *generate* vehicle speed (Van Brummelen, O'Brien, Gruyer, & Najjaran, 2018). We propose adding three AI-related concepts to Brennan and Resnick's CT framework: *Classification*, *Prediction*, and *Generation*.

Classification. Machines often sort information into categories for downstream decision-making through rules (e.g., "the sentence is positive because it contains 'happy'") or learning algorithms (e.g., after observing "positive" sentences, similar sentences are classified as "positive").

Prediction. To act intelligently, machines predict future values and behavior. This includes predicting the category an object may fall into, an object's future behavior, or the best action to take next (e.g., after saying, "I am a", a conversational agent may predict the next best word to be "robot").

Generation. Using information gathered, machines can generate new data. This may include synthesizing previous examples, creating new information, or making decisions (e.g., a machine constructing and speaking a new sentence**).**

We also propose the following AI-related practice:

Training, Validating, and Testing. Developing a robust ML model requires waiting for the model to learn to recognize patterns, testing if it generates correct predictions, and determining if it is sufficient for the task. Training involves providing examples (or an environment) for the model to iteratively learn from (or experiment in). Testing and validating involves providing different examples (or environments) to observe how the model behaves, comparing the model's behavior to other models, and determining whether the model is sufficient. This includes assessing the *accuracy* of the model (e.g., the percentage of correct classifications) using a test and/or validation dataset and the model *loss* (a value used to update model weights during training).

Finally, we propose the following AI-related perspectives:

Evaluation. Some AI (e.g., neural networks and other learning techniques) behavior can be difficult to predict or unintuitive to humans. Programmers must think about how well the program behaves and whether it achieves the necessary goals (e.g., How can we improve the program? Did we over- or under-train the model? Is the model biased towards certain people because it was trained by them?). These considerations are especially pertinent when considering the large number of input-output relationships with ML.

Evaluation is performed in the context of the final product or application, whereas testing and validating are performed only considering the model itself. For example, during evaluation, one might ask, "Is my app biased towards classifying people as middle-aged?"; whereas during validation, one may ask, "Why is my model achieving 42% accuracy?".

3. CONVERSATIONAL AI EXTENSION

The conversational AI extension for MIT App Inventor, or *Text Mixer*, generates text based on three input corpora: Dr. Seuss books, Taylor Swift lyrics, and Shakespearean poetry. It enables students to generate text resembling the input corpora by providing corpora weights (*mixture coefficients)*. The model contains three, single-layer LSTM models with 30 hidden units and static GloVe embeddings (*6B*, *d=300*) (Pennington, Socher, & Manning, 2014).

- **SEUSS**: The Dr. Seuss collection of children's poetry contains word coverage representative of children's literature at a K-3 reading level (Foster & Mackie, 2013).
- **SWIFT**: Lyrics from the popular artist Taylor Swift contain colloquialisms, interjections, and repetitions and focus on the themes of love and heartbreak from an adolescent perspective (Kotarba, 2013).
- **SHAKESPEARE**: These works are featured in university and high school English courses, and consist of Early Modern English verses (G. T. Wright, 1983).

To generate response-sentences from a mixture of language models, we resampled the model at each word-generation step. For each word, we sample from the three models based on the mixture coefficients inputted to the block (See Figure 1). Using the sampled model, we feed in the input sequence of previously generated words until an end-of-sentence token or the maximum length has been reached. Upon completion, we have both a sequence of newly generated words and the corresponding list of language models each word was generated from.

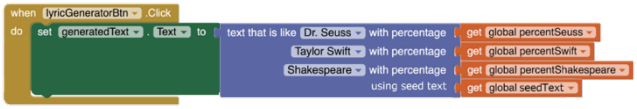

Figure 1. Example Text Mixer block in MIT App Inventor

4. RELATION TO CT FRAMEWORK

With the conversational AI *Text Mixer* extension, students can explore the proposed AI concepts, practice, and perspective.

Classification. In the Duet App, the *Text Mixer* extension contains ML models to classify and organize words in latent spaces (the representation spaces where neural networks organize information). The models use this organization to determine words' likelihood to appear next in the sentence.

Prediction. The *Text Mixer* extension predicts the best next word in the sentence by using three ML language models. The mixture coefficients determine how often a model is chosen (e.g., if the Dr. Seuss mixture coefficient is relatively high, then the Dr. Seuss language model will likely be chosen). The ML language models then use the "seed text", the previous words in the sentence being generated (if any), and previously-trained weights to predict the best next word.

Generation. After predicting a word, the *Text Mixer* adds this word to the sentence. This repeats until a full sentence is generated. Once a sentence has been constructed, the *Duet App* speaks the words aloud while playing music.

Training, Validating, and Testing. The *Text Mixer* block exemplifies training by enabling students to choose machine learning models trained on different input corpora. This block is meant to be a high-level introduction to ML, so it does not necessarily exemplify testing and validating. The authors plan to develop blocks for testing and validation in future work.

Evaluation. After developing the app, students can evaluate the output to determine whether the model generates song lyrics adequate for their application.

5. CONCLUSION

CT frameworks need to be updated to continue to be relevant in an increasingly AI-powered world. This paper proposes AI-related CT concepts, practice, and perspective, building on Brennan and Resnick's (2012) framework and presents an MIT App Inventor AI extension. The skills learned through experimenting with the *Text Mixer* integrate smoothly with the CT framework and help students to better understand AI.

6. REFERENCES

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 annual meeting of the American Educational Research Association*. Vancouver: American Educational Research Association.

Foster, J., & Mackie, C. (2013). Lexical analysis of the Dr. Seuss Corpus. *Concordia Working Papers in Applied Linguistics, 4,* 1-21.

Kotarba, J. A. (2013). *Understanding Society through Popular Music*. New York: Routledge.

Malyn-Smith, J., Lee, I. A, Martin, F., Grover, S., Evans, M. A, & Pillai, S. (2018). Developing a framework for computational thinking from a disciplinary perspective. *Proceedings of the International Conference on Computational Thinking in Education 2018*. Hong Kong: The Education University of Hong Kong, 182-186.

MIT App Inventor. (2017). *Anyone Can Build Apps That Impact the World.* Retrieved January 24, 2019, from http://appinventor.mit.edu/explore/

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). Association for Computational Linguistics,* 1532–1543.

Van Brummelen, J., O'Brien, M., Gruyer, D., & Najjaran, H. (2018). Autonomous vehicle perception: The technology of today and tomorrow. *Transportation Research Part C: Emerging Technologies, 89,* 384-406.

Winston, P. H., & Shellard, S. A. (1990). *Artificial intelligence at MIT: expanding frontiers*. MIT Press.

Wright, G. T. (1983). The play of phrase and line In Shakespeare's iambic pentameter. *Shakespeare Quarterly*, *34*(2), 147-158