AI Snap! blocks for speech input and output, computer vision, word embeddings, and neural net creation, training, and use

Ken Kahn, **University of Oxford, toontalk@gmail.com**

Naveen Naveen, AI World School, naveen@robotixedu.com

Ramana Prasad, AI World School, rp@robotixedu.com

Gayathri Veera, AI World School, gayathriveera@merituseducation.com

Abstract

We will demonstrate blocks integrated into Snap! (Harvey, B., & Mönig, J. 2010) (Snap 2021) capable of a wide range of AI services, interactive AI programming guides, and a selection from over thirty sample projects. These blocks have been used by school students as young as 6 and by university students in both school settings and informal learning contexts. We are aware of sessions and workshops that have been held in many European, Asian, and North American countries.

The Snap! AI library includes blocks for speech synthesis and recognition. These rely upon standard web services currently provided by Chrome and Edge. Other blocks access computer vision cloud services provided by Google and Microsoft.

Relying upon the open source TensorFlow.js library (Smilkov et al 2019) are blocks that load and use pre-trained neural network models. These blocks perform image classification, object detection and segmentation, pose detection, style transfer, sentence and image encoding, and sound recognition. Other blocks rely upon TensorFlow.js to support neural network creation, training, prediction, and hyper-parameter optimization. These blocks run on the user's devices thereby avoiding application installation, latency from network access, and possible privacy violations from data being sent to servers. Finally, Snap! blocks relying upon TensorFlow.js can be run offline, enabling their use in parts of the world lacking reliable Internet connections.

There are blocks that support word embeddings by reporting the embeddings, finding nearest words, and mapping words to two-dimensional locations. They support 20,000 words and 15 languages.

In addition to the library of Snap! blocks we will demonstrate the AI programming guide that contains a large number of interactive elements designed to enable students to explore the Snap! blocks in the context in which they are presented.

Our demonstration will include a range of sample projects that illustrate the use of the AI blocks library. These projects were designed to be relatively easy for students to modify and enhance.

An introduction to a wide sample of the Snap! AI blocks

A general principal followed in the design of the Snap! AI blocks is to provide at least two versions for each functionality. One version is as simple as possible while the other provides a large variety of options and advanced functionality. For example, two of the blocks for speech synthesis follow. (Note that it is up to the browser's implementation of each synthesis voice how, as below, a French voice is given English text. In Chrome it speaks the English with a French accident and speaks numbers in French.)

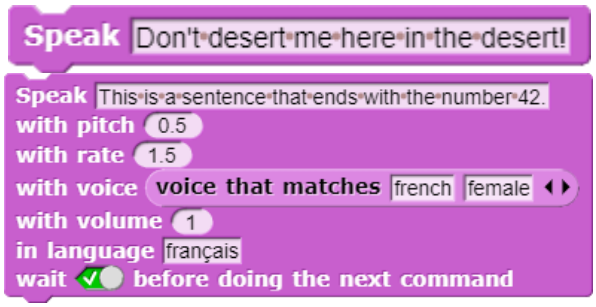

Figure 1 – Two blocks for speech synthesis

Determining good defaults for the simple blocks is challenging. Should the simple speak block begin speaking and then immediately proceed to the next command or proceed only when finished speaking? We chose to have it not wait without much knowledge of how most users will use it. The situation is made more difficult by the fact that changing the default later would likely lead to confusion and compatibility problems. And this burden would fall on the novice users who is yet to master the expert version of the command.

A different sort of challenge arose with the block that reports a voice that can be used in the advanced speech synthesis block.

Figure 2 – a block to find the desired voice

Each browser has a different list of available voices. And the list can differ in different locales. The "voice that matches" block searches through the available voices for one that best matches the keywords. The block in Figure 2 reports a female English speaking voice with a British accent in Chrome (at least in some parts of the world) but in other browsers may report a male US English voice if nothing closer is available.

Full-featured speech synthesis and recognition blocks support an optional language specification. The default language is the default language of the browser being used. A block can set it to another value. Effort was taken to enable users to specify the language in English, in the language specified, or as a IEFT language tag (e.g., fr-FR for French as spoken in France).

Speech recognition blocks are also available in simple and full-featured forms. The simplest block waits until the browser reports what was just spoken (or an error message if nothing was spoken or the microphone was not accessible).

Figure 3 - A block that reports the next result from the speech recognizer

A more advanced user is more likely to find the following block more useful. It will listen in the background and when the speech is recognized a user-supplied script is run.

Figure 4 - This block will run the first script with text of the recognized speech or the second script with an error message

For very advanced usage of speech recognition access to alternative interpretations of the speech and the confidence scores of each can be achieved with this block.

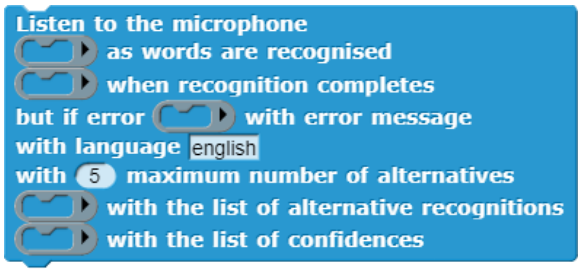

Figure 5 – full featured speech recognition block

We developed a set of blocks to access AI cloud services for image recognition and the like. These blocks report a detailed analysis of an image describing categories, objects, bodies, and much more. We provide blocks for accessing any part of the analysis as well as blocks for convenience to access commonly used features such as labels. The blocks can access services from Google, Microsoft, and originally also IBM (but they changed their API making this infeasible). A significant awkwardness using these services is that one needs to register to obtain an API key. Free quotas are generous but do limit the number of queries one can make per time period. Students need to be careful to not reveal their API keys to others. These blocks were rarely used by students.

Fortunately, subsequent to the development of those blocks, TensorFlow.js pre-trained models for image recognition, object detection, body segmentation, and pose detection became available. There are blocks that provide easy-to-use access to each of these functionalities. They can work either with the user's device's camera or with stored images. While not as accurate or elaborate as the analysis provided by AI cloud services, they work very quickly without sending any data off the device.

Inspired by the first version of Teachable Machine (Google 2021), we developed blocks that enables one to train a model to classify images into user-defined categories. Subsequently, version 2 of Teachable Machine was released that enabled us to provide blocks that can use inside Snap! a model trained in Teachable Machine to categorize images or sounds. Teachable Machine has a very well-engineered user interface and training functionality that is better than what the Snap! blocks can provide. However, the Snap! blocks can be used in applications where a model is trained, used, and then more training and use follows.

Figure 6 – A block for getting predictions from a model trained in Teachable Machine

Included in the Snap! AI block library are blocks for doing natural language processing. These include blocks that support word embeddings in 15 languages, English sentence embeddings, and question answering from a passage. An example of using these blocks is solving word analogy problems such as "man is to father as woman is to X".

Figure 7 – using simple version of word embedding blocks to solve a word analogy problem

The final set of Snap! AI blocks support the creation, training, and use of deep neural networks. Once again the simple and full-featured versions are provided. A model can be created with either of the following blocks.

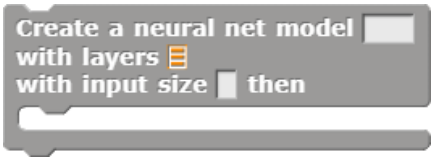

Figure 8 – a simple block for creating a deep neural network

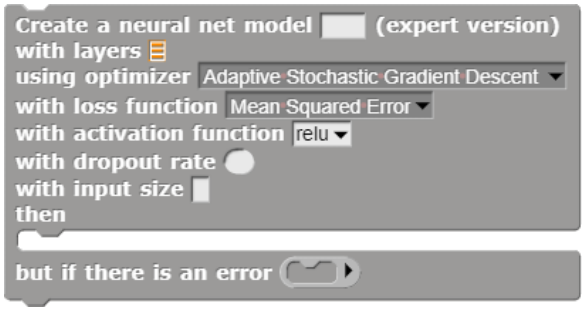

Figure 9 – the full-featured block for creating neural networks

Interactive elements in the programming guide

A tutorial guide to all the Snap! AI blocks is provided as a web site. The tutorial material is interleaved with discussions of societal impact, history, background material, project suggestions, and more. Each section of the guide contains several frames that provide a means to interact with the Snap! blocks discussed directly without leaving the page and its guidance.

A sample of AI Snap! projects

We have developed thirty sample artificial intelligence programs in a form suitable for enhancement by non-expert programmers. The projects are implemented using the Snap! AI blocks. These projects have been designed to be modifiable by school students and have been iteratively developed with over 100 students. The projects involve speech synthesis, speech and image recognition, natural language processing, and deep machine learning. They illustrate a variety of AI capabilities, concepts, and techniques. The intent is to provide students with hands-on experience with AI programming so they come to understand the possibilities, problems, strengths, and weaknesses of AI today. (Kahn & Winters 2021b)

Student experiences using the Snap! AI blocks and associated resources

The Snap! AI blocks have been used by school students as young as 6 and by university students in both school settings and informal learning contexts. We are aware of sessions and workshops that have been held in many European, Asian, and North American countries. Publications describe some of these efforts in Indonesia (Kahn et al 2018) and China (Kahn et al 2020).

Here we describe previously unpublished descriptions of some student experiences in India.

We carried out a pilot program with AI Education courses curated for K-12 children. Scratch for AI blocks and Snap! for AI blocks were recently introduced to students in India to help them deduce and understand various AI functions by creating programs which gave them access to real time AI services. The curriculum and course for the pilot program were created under the guidelines as outlined by the Artificial Intelligence (AI) for K-12 [AI4K12] initiative (https://ai4k12.org/).

In the first phase of the program, since children in India are familiar with the Scratch programming platform (MIT Media Lab), we introduced Scratch for AI to children between the ages 8 to 13. With Scratch for AI, children could explore 12 different types of functions to AI services which include Computer Vision, Web access and retrieval, Speech Synthesis, Language Recognition, Machine Learning etc. Children could understand a complex topic like Machine Learning to train the computer to identify different objects and even go on to create a complex program using the Posenet function.

In the second phase of the program, in order to introduce the children to higher level programming, like first class lists and first class procedures, and to advanced AI functionalities Snap! for AI blocks programming was a logical choice especially for those in their teenage years. Children between the ages 14 to 18 were taught how to create programs with AI functionalities using the Snap! for AI blocks. One example of a program found popular by the children was where they used computer vision to help detect whether the user is wearing a mask or not, attributing it to a real world situation. Another program that was quite popular with the children was where neural-style transfer effects were added to images.

From our initial observations of the children's experiences creating their own programs with the Scratch for AI and Snap! for AI blocks, we found that the children were able to learn and explore AI in fun ways, and they also assumed greater responsibility and autonomy for their interest in learning about AI.

Conclusions

We continue to develop Snap! blocks and resources to provide beginner-friendly access to AI functionalities. We hope to give learners and hobbyists the ability to creatively use AI in their projects thereby acquiring a first-hand understanding of some elements of a technology that is rapidly changing our world.

References

eCraft2Learn 2021. A guide to AI extensions to Snap!. https://ecraft2learn.github.io/ai/ Accessed: 07-09-2021.

Google 2021. Teachable Machine. https://teachablemachine.withgoogle.com/ Accessed: 2021-09-07.

Harvey, B., & Mönig, J., 2010. Bringing "No Ceiling" to Scratch: Can One Language Serve Kids and Computer Scientists? In Proceedings: Constructionism, Paris, France.

Kahn, K.M. and Winters, N., 2018. AI programming by children. In Proceedings: Constructionism, Vilnius, Lithuania.

Kahn, K.M., Megasari, R., Piantari, E. and Junaeti, E., 2018. AI programming by children using snap! block programming in a developing country. Proceedings: EC-TEL. Leeds, UK.

Kahn, K., Lu, Y., Zhang, J., Winters, N. and Gao, M., 2020. Deep learning programming by all. Proceedings: Constructionism, Dublin, Ireland.

Kahn, K.M. and Winters, N., 2021a. Constructionism and AI: A history and possible futures. British Journal of Educational Technology.

Kahn, K. and Winters, N., 2021b. Learning by Enhancing Half-Baked AI Projects. KI-Künstliche Intelligenz, pp.1-5.

Smilkov, D., Thorat, N., Assogba, Y., Yuan, A., Kreeger, N., Yu, P., Zhang, K., Cai, S., Nielsen, E., Soergel, D. and Bileschi, S., 2019. Tensorflow. js: Machine learning for the web and beyond. arXiv:1901.05350.

Snap 2021. Snap! Build your own blocks. https://snap.berkeley.edu/ Accessed: 2021-08-23.