**Title**: Fun projects with language translation

**Summary:** Natural language processing is growing in importance. We often converse with automatic chatbots for customer service without even knowing. We also use online translation services or mobile apps. But how do these services work? Is there artificial intelligence (AI) in them? Three projects are offered to cater for student interest and different programming skill levels.

This lesson was developed in collaboration with the Digital Technologies Institute.

**Year Levels**: 3–4, 5–6

Preliminary notes

AI specific tools used in the plugged part of the activity:  [Google Translate](https://translate.google.com/?hl=en&tab=TT&authuser=0) and AI Chatbot [Mitsuku](http://www.mitsuku.com).

Suggested steps

Unplugged activities

The purpose of this activity is for students to identify general patterns of communication in the form of question and answer.

1. Use [inside/outside circles](http://www.theteachertoolkit.com/index.php/tool/inside-outside-circles) to enable students to have a conversation between two students. One student in the pair takes on the role as a visitor from overseas; the other is asks questions and finds out about the visitor. Students have three minutes to find out as much as possible about their travels. After three minutes, the outside circle of students moves in a clockwise direction, five people along. Repeat at least 2–3 times.
Optional: The student playing the visitor may answer the questions in the language being studied.
2. In pairs, taking on the same role as in the previous task, students have a conversation. This time, however, the student playing the visitor has to anticipate the questions they think they will be asked.
* The student finding out about the visitor will write six questions on a strip of paper without revealing them to their partner.
* The student playing the visitor will write down six responses on a strip of paper. They will also write down a separate response (in case a question is not one they anticipated) such as, ‘I’m not sure’, or I’ll have to think about that.’

For example, anticipating the questions might look something like:

**If**the question is about finding out my name **then … my response is …**

**If**the question is about where I come from **then … my response is …**

**If**the question is about what I am doing **then … my response is …**

**If**the question is asking if I’m lost or need help **then … my response is …**

**If**the question is about what I have liked **then … my response is …**

**If**the question is about where I’m going next **then … my response is …**

**else … my response is …**I’ll have to think about that.’

1. The conversation plays out as the questions are put forward and the best matching response is presented. Students tally the number of responses that were a good match to the question.

This task is in preparation for creating an algorithm and implementing a computer program to create a chatbot that responds to input and also translates the output into another language and speaks it.

Inclusive Education variations

####  Having limited, low or no vision

**Students with a vision impairment** would benefit from adjustments to make the tasks accessible, for example students with:

* low vision would benefit from larger font for words written on paper strips
* low contrast vision may benefit from acetate or colour filters placed over on paper strips
* no vision may benefit from words on paper strips with braille-embossed print

####  Deaf or hard of hearing

**Students with a hearing impairment** would benefit from adjustments to make the tasks accessible, for example:

* Students could use hearing assistive technology systems (HATS) such as an FM system or similar sound-field system where the teacher talks into a microphone and their voice is distributed around the classroom.
* Teachers may also use closed captioning on any programming instructional videos, provide transcripts and repeat instructions, as well as having the student sit in close proximity to them.

**Additional Scaffolding/Limited Abstract Thinking Skills:**

If students find this activity challenging, you can modify it so that students just work in pairs and/or instead of the repeating the same roles, students switch roles (afterwards the teacher can lead a discussion about whether it was easier to ask questions second).

Plugged activities

These activities are organised in terms of levels starting from the simplest activity and progressively getting more challenging.

Mitsuku (Skill level: easy)

After you have introduced the concept of Artificial Intelligence, you can then demonstrate a chatbot, Mitsuku, to them. Mitsuku can be found at <https://www.pandorabots.com/mitsuku/> and Google Chrome browser is recommended for best results. Students can also practise asking questions to Mitsuku on their own devices.

It is important that students understand that a computer is doing the work of interpreting what the user is typing and making a guess as to how to respond, even if it seems like Mitsuku is a real person.

For students who need assistance with typing and/or spelling, you can use the microphone and predictive text features on laptops and tablets so students can communicate with Mitsuku without typing.

Google Translator (Skill level: easy)

Google Translator is a good example of a computer being able to take an input and not just respond to it, but to do work between the input and the response (like a calculator). This is one of the many reasons we use computers, to help us do work, such as translating a word from one language to another. Computers can do this much faster than a person with a dictionary.

1. Explore translating text and speech into a language of your choice using Google Translate on a digital device enabled with a microphone and camera.

Use the Detect language feature of Google Translate to translate into the required language. This feature uses AI to translate using speech recorded with a microphone; or text in the form of typed words and phrases; or camera shots of selected text. The ‘Translate by Voice’ feature in Google Chrome where students speak a word into the microphone and the browser speaks the translation back.

You can use this version of Google Translate in your browser: https://translate.google.com/

For Android devices use Google Play to download the Google Translate app or for iOS devices go to App Store and download Google Translate .

1. Make a list of the words and phrases that you attempted to get the app to translate.

What was the success rate of the App in providing a correct response to your input?

What were some limitations of the App?

Text to speech translator (Skill level: easy)

Use Scratch 3.0 and the blocks of ‘Text to speech’ and ‘Translate’. To add these blocks, click on the blue ‘Add extension’ box on the bottom left of the Scratch screen. In this program students can explore text to speech blocks and translating into different languages.

**Additional Scaffolding/Limited Abstract Thinking Skills:**

You can either have students build the program from scratch (pun intended!) or if they find the task challenging, students can look at the completed code first to learn how it works. It can be found here: <https://scratch.mit.edu/projects/338489965/editor/>

1. The first step is to create a simple program that uses an ‘Ask’ block to ask the user what they want to translate. Test the program to make sure the answer is displayed on screen by the sprite.



*Image: Scratch program showing Sprite 1 coding blocks to input text*

1. The next step is to add the translate block to replace the answer.
2. Change the language to your choice of language.
3. The answer block replaces the ‘hello’ text in the translate block.

The text on screen will now be in the language chosen.



*Image: Scratch program showing Sprite 1 coding blocks to translate input text*

1. The final step is to get the answer spoken in the translated language.
* Add the speak block.



* Replace the hello text with the translate block.
* Replace the hello text of that block with the answer block.

So now when you type in the input bar, the text (answer) is translated in text on screen, and then spoken in the selected language.



*Image: Scratch program showing Sprite 1 coding blocks to translate input text and speak translated answer*

*Some ideas for remix*

1. Change the language.
2. Change sprite and background.
3. Explore and build your own program.
4. Create your own code block to enable the user to change language to another language.

Sample code: [Text to speech translator](https://scratch.mit.edu/projects/338489965/)

Chatbot with language translation (Skill level: medium)

Use Scratch 3.0 and the additional blocks of ‘Text to speech’ and ‘Translate’.

1. Create a chatbot to chat with a visitor from overseas. In coding this program, text inputs (a string of text) are referred to as the ‘answer’. The chatbot responds if the ‘answer’ contains a particular word, or else a generic response is given. The more words that are coded with a relevant response, the higher the chance a good match will be made with the questions input by the user. The program runs forever. [Note: in the program the user is actually asking the question but the program **deals with this as the ‘answer’**, just a limitation of Scratch and its logic.]

Two sample programs have been provided to help scaffold the activity:

* [Foreign Chat](https://scratch.mit.edu/projects/338464637/)
* [Foreign Chat basic\_1 Question ONLY](https://scratch.mit.edu/projects/338483842/editor)
1. It may be useful for students to explore the fully functional program [Foreign Chat](https://scratch.mit.edu/projects/338464637/) first to input their question and see how the chatbot responds. Students could rate the chatbot on how many of their questions it was able to provide with a ‘smart’ response.
2. A basic code [Foreign Chat basic\_1 Question ONLY](https://scratch.mit.edu/projects/338483842/editor)  is provided for students to remix. In remixing the project students could:
* change the language
* change the sprite and background
* change the word contained in the answer and test
* add new **if/else** blocks and duplicate the existing **if** code block. (To do this nest the new **if/else** block in the previous **else** option. Continue this process for each additional block until the last **else** has the generic answer for **no match found**.)



*Image: Scratch program showing Sprite (Abbey) coding blocks to respond to input text in selected language*

Discuss:

* In what ways is the program smart?
* What are some limitations of the program?
* How well did the translation work?

Automating a conversation (Skill level: med to hard)

1. Use Scratch 3.0 and the additional blocks of ‘Text to speech’ and ‘Translate’ to program a conversation between two people.  Try to make the program as smart as possible.

Note: this project does not include user input. Instead, it focuses on automating a conversation. For a focus on a program that requires user input consider chatbot or translate projects as shown above.

In this program skills required include:

* **Select sprites** (characters) and program with **say** blocks.
* **Randomise** responses using **Random** function (stored as a variable).
* **Create, store and use a variable**, make a list to store phrases.
* Use **broadcast message** to trigger responses.
* Use **loops** to continue the conversation.
* Use ‘**text to speech**’ to read all conversation text.
* Use **translate** to translate one sprite’s responses in the language being studied.



*Image: Scratch program showing Sprite 1 coding blocks*

**

*Image: Scratch program showing part of Sprite 2 (Abby) coding blocks*

1. Use the sample code to explore text to speech and translate functions in Scratch. Play the program and then look inside to reveal how it was coded.

**Sample code**: Random convo: [Translate (German)](https://scratch.mit.edu/projects/337833620/)

*Some ideas for remix:*

* This example translates to German but you could remix in another language.
* Change greetings and convo starters and responses stored as lists.
* Work out a way not to repeat phrases used in the conversation.
* Create your own conversations using your own sprites.

Exploring AI [Skill level: easy]

1. Explore translating text and speech into a language of your choice using Google Translate on a digital device enabled with a microphone and camera.

Use the Detect language feature of Google Translate to translate into the required language. This feature uses AI to translate using speech recorded with a microphone; or text in the form of typed words and phrases; or camera shots of selected text.

You can use this version of Google Translate in your browser: <https://translate.google.com/>

For Android devices use Google Play to download the [Google Translate app](https://play.google.com/store/apps/details?id=com.google.android.apps.translate&hl=en_AU) or for iOS devices go to App Store and download [Google Translate](https://apps.apple.com/au/app/google-translate/id414706506).

1. Make a list of the words and phrases that you attempted to get the app to translate.
* What was the success rate of the App in providing a correct response to your input?
* What were some limitations of the App?
1. In addition, you could explore the AI chatbot aspect with [Mitsuku](http://www.mitsuku.com).

Discussion

1. Discuss the ways in which the AI in Google Translate responded to your input (text, audio and image).
* Did you notice any issues depending on your input?
* How might the AI be useful in day-to-day life?
1. Compare your programming version using Scratch with your exploration of Google Translate.
* What is similar and what is different?
1. Discuss how websites often use chatbots for ‘triage’ before the conversation is handed over to a human customer service agent.
* Why do organisations use AI chatbots?
* What are some benefits?
* What are some potential issues?

Why is this relevant?

Algorithms and programming are an essential aspect of developing machines that are powered by Artificial Intelligence (AI). AI is the ability of machines to mimic human capabilities in a way that we would consider 'smart'.

In conventional programming, the computer is provided with a set of instructions for a defined set of scenarios. In our conversation program the students hard coded the program.

To mimic the AI, we used the random function in ‘Automating a conversation’ program to come up with random phrases to which a ‘smart’ response is given. In all three example programming projects we incorporated Natural Language Processing (NLP). NLP is the ability of machines to interpret and analyse forms of human communication, such as text and speech. In our programs all responses where hard coded. In true AI, NLP aims to mimic human communication by teaching the machine to read, write, speak and listen by providing it lots of examples of communication data and teaching the machine by example or by letting it discover patterns on its own.

Machine Learning is an application of AI. With Machine Learning, we give the machine lots of examples of data, demonstrating what we would like it to do so that it can figure out how to achieve a goal on its own. The machine learns and adapts its strategy to achieve this goal.

In the Google Translate example, the machine would have been fed enormous amounts of data on languages that included both text and speech. The AI has learned to respond to recognise the input and provide a suggested translation.

This lesson focuses on:

* representation of words as text and audio (including in other languages)
* branching and iteration
* user input
* recognition of words as text, audio and as an image
* the difference between explicit programming and learning by example as in AI.

Resources

* Article: [Google’s live camera translation is getting better AI and 60 new languages](https://www.theverge.com/2019/7/10/20688682/google-translate-camera-instant-translation-function-languages-update-ai)
* AI chatbot: [Mitsuku](http://www.mitsuku.com)

Assessment

Teacher assessment

From these suggestions, choose the task and assessment approaches that will best suit your students.

|  |  |  |
| --- | --- | --- |
| **Task skill level** | **Tasks** | **AC content descriptions** |
| EasyMediumMedium to hardAI (easy) | Describe how words were represented in different formats such as text and audio. Demonstrate how speech was translated into another language in your programming.  | ACTDIK008ACTDIK014 |
| EasyMediumMedium to hard | Describe the hardware and software used in making your conversation program. Explain how it works. What data is used and stored? How did you design and create your user interface?  | ACTDIK007ACTDIK014ACTDIP010ACTDIP018ACTDIP019ACTDIP020 |
| AI (easy)EasyMediumMedium to hard | Explain how you used Google Translate or similar translation program.* Describe how the AI responds to different inputs.
* Compare the AI to your programmed conversation. How smart was your programmed version?
* Explain how your solution meets the needs of the user.
 | [ACTDIK014](http://www.australiancurriculum.edu.au/curriculum/contentdescription/ACTDIK001)[ACTDIK023)](http://www.scootle.edu.au/ec/search?accContentId=ACTDIK007)ACTDIP012ACTDIP021 |
| AI (easy)EasyMediumMed to hard | How might this type of AI used in our daily lives? | ACTDIP021ACTDIP031 |

Australian Curriculum Alignment

Technologies – Digital Technologies

**Years 3–4**

Identify and explore a range of digital systems with peripheral devices for different purposes, and transmit different types of data (ACTDIK007)

Recognise different types of data and explore how the same data can be represented in different ways (ACTDIK008)

Define simple problems, and describe and follow a sequence of steps and decisions (algorithms) needed to solve them (ACTDIP010)

Implement simple digital solutions as visual programs with algorithms involving branching (decisions) and user input (ACTDIP011)

Explain how student solutions and existing information systems meet common personal, school or community needs (ACTDIP012)

**Years 5–6**

Examine the main components of common digital systems and how they may connect together to form networks to transmit data (ACTDIK014)

Design a user interface for a digital system (ACTDIP018)

Design, modify and follow simple algorithms involving sequences of steps, branching, and iteration (repetition) (ACTDIP019)

Implement digital solutions as simple visual programs involving branching, iteration (repetition), and user input (ACTDIP020)

Explain how student solutions and existing information systems are sustainable and meet current and future local community needs (ACTDIP021)

Languages: German

Note: content descriptions will depend on the language students are using.

|  |  |  |
| --- | --- | --- |
| **Years 3–4** |  | Experiment with the pronunciation of vowel sounds, letter combinations and intonation patterns, and recognise and write high-frequency words and expressions in familiar contexts[Key concepts: pronunciation, intonation, accents; Key processes: distinguishing sounds, recognising, practising] (ACLGEU131) |
| **Years 5–6** |  | Explain and apply basic rules for German pronunciation, intonation, spelling and punctuation[Key concepts: pronunciation, writing systems, punctuation; Key processes: spelling, making connections, applying rules] (ACLGEU148 ) |