Discursis: Conversation analysis and visualization

Introduction

The Discursis tool is an open-source implementation of the previous Discursis software. The tool is designed to be used in Jupyter notebooks, alongside other tools from the Australian Text Analytics Platform (ATAP) project. Discursis was developed by the Sydney Informatics Hub (SIH) as part of ATAP.

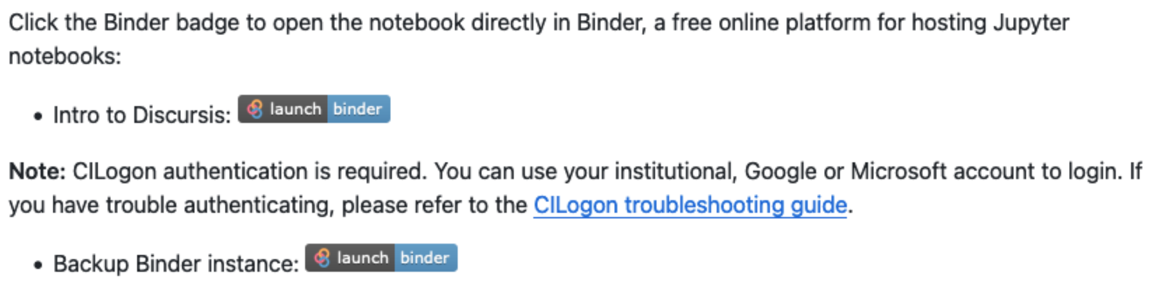
The tool is designed to analyse and visualise the recurrence of topics across a conversation (or similarly structured data, e.g. social media threads), as described in Angus (2012). The tool produces an overall visualisation of this recurrence across the conversation, as well as exports of the different metrics summarizing this recurrence.

Getting started

You can try out the Discursis tool online in a Jupyter notebook, via Binder. You'll find links to launch Binder on the GitHub page for Discursis.

If you're not familiar with Jupyter Notebooks, you can find a short introduction to them here.

You can launch a demonstration of Discursis from the GitHub page by clicking one of the 'launch Binder' buttons.



Click the Binder badge to open the notebook directly in Binder, a free online platform for hosting Jupyter notebooks:

- Intro to Discursis: launch binder

Note: CILogon authentication is required. You can use your institutional, Google or Microsoft account to login. If you have trouble authenticating, please refer to the CILogon troubleshooting guide.

- Backup Binder instance: launch binder

If you are a university student or educator, you can access the ATAP Binderhub via CILogon. CILogon supports the single sign-on (SSO) method with most (Australian or international) institutional login credentials, or with a Google/Microsoft account.

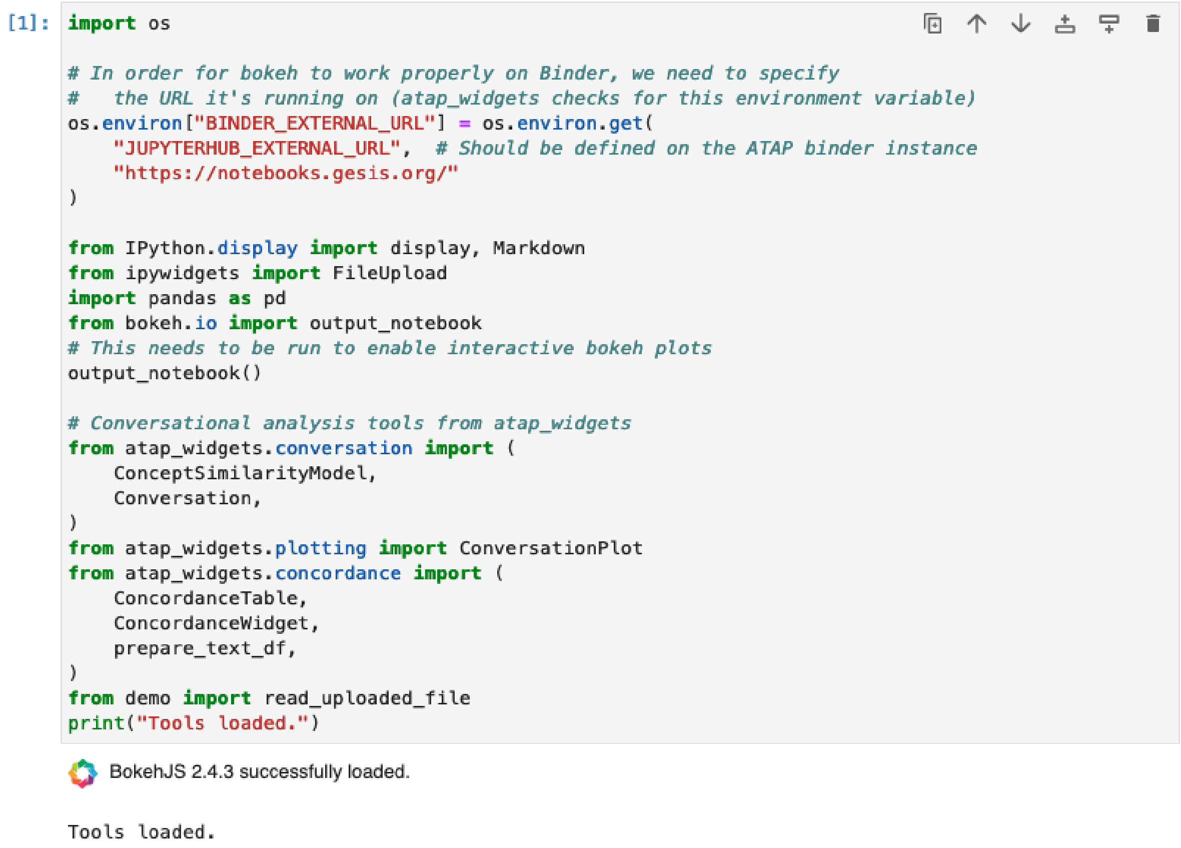
If you can't access via CILogon, you can click the button next to "Backup Binder instance" to use a free version of Binder.

Using Discursis

Launch Binder as described above. It should launch directly into the demo notebook (after a short wait). The demo notebook has been set up so that you can run the Discursis tool in a small number of steps.

Load Python tools

Select and execute the notebook cell under "Python setup" to load the different Python tools used in the demo. You should see a message saying "Tools loaded" once this is done, as shown:

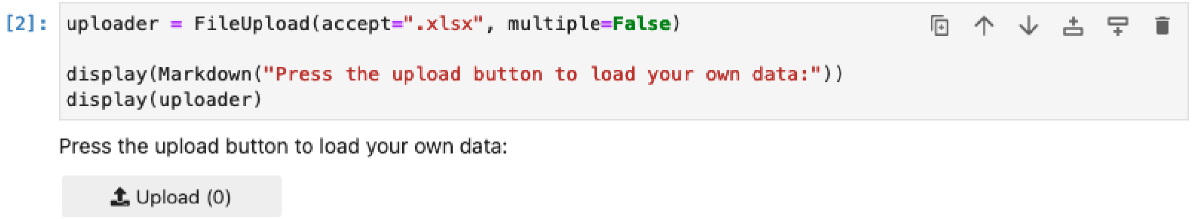


```
[1]: import os

     # In order for bokeh to work properly on Binder, we need to specify
     #   the URL it's running on (atap_widgets checks for this environment variable)
     os.environ["BINDER_EXTERNAL_URL"] = os.environ.get(
         "JUPYTERHUB_EXTERNAL_URL",  # Should be defined on the ATAP binder instance
         "https://notebooks.gesis.org/"
     )

     from IPython.display import display, Markdown
     from ipywidgets import FileUpload
     import pandas as pd
     from bokeh.io import output_notebook
     # This needs to be run to enable interactive bokeh plots
     output_notebook()

     # Conversational analysis tools from atap_widgets
     from atap_widgets.conversation import (
         ConceptSimilarityModel,
         Conversation,
     )
     from atap_widgets.plotting import ConversationPlot
     from atap_widgets.concordance import (
         ConcordanceTable,
         ConcordanceWidget,
         prepare_text_df,
     )
     from demo import read_uploaded_file
     print("Tools loaded.")

     BokehJS 2.4.3 successfully loaded.

     Tools loaded.
```

Upload data

In order to support using the Discursis tool on your own dataset, the notebook expects you to upload an Excel file containing the conversation data to analyse. The spreadsheet should contain at least two columns: **text**, containing the text of each utterance in the conversation, and **speaker**, identifying the speaker for the utterance, with one row per utterance. The spreadsheet can contain other columns. If you just want to test out Discursis, you can use this spreadsheet, containing the transcript of the National Press Club Leaders Debate between Kevin Rudd and Tony Abbott, available at the Parliament of Australia website under a CC BY-NC-ND 3.0 AU Creative Commons license.

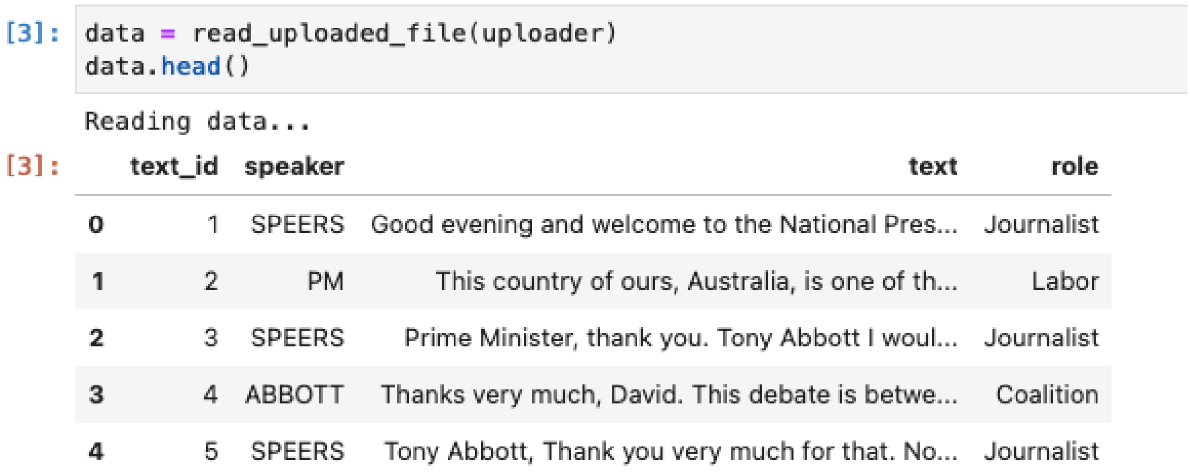
To upload the spreadsheet, first execute the cell under "Upload data" – an upload button should appear. Click the "Upload" button and select your spreadsheet. Once done, you should see "(1)" appear in the upload button, indicating 1 file has been uploaded.



```
[2]: uploader = FileUpload(accept=".xlsx", multiple=False)

     display(Markdown("Press the upload button to load your own data:"))
     display(uploader)
```

Press the upload button to load your own data:

After uploading, execute the following cell. You should see a preview of the first few rows of the uploaded spreadsheet:



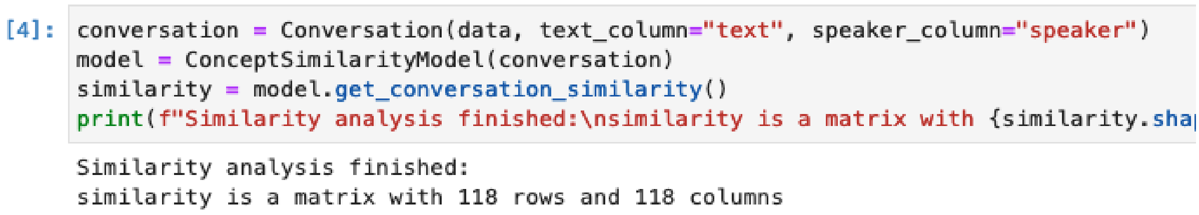
```
[3]: data = read_uploaded_file(uploader)
     data.head()

     Reading data...
```

[3]:

	text_id	speaker	text	role
0	1	SPEERS	Good evening and welcome to the National Pres...	Journalist
1	2	PM	This country of ours, Australia, is one of th...	Labor
2	3	SPEERS	Prime Minister, thank you. Tony Abbott I woul...	Journalist
3	4	ABBOTT	Thanks very much, David. This debate is betwe...	Coalition
4	5	SPEERS	Tony Abbott, Thank you very much for that. No...	Journalist

Analyse conversation

Execute the next code cell to perform the conversational analysis on the data, using the method from Angus et al. (2012). After executing, you should see a message showing that the analysis is finished and a similarity matrix has been produced:



```
[4]: conversation = Conversation(data, text_column="text", speaker_column="speaker")
     model = ConceptSimilarityModel(conversation)
     similarity = model.get_conversation_similarity()
     print(f"Similarity analysis finished:\nsimilarity is a matrix with {similarity.sha

     Similarity analysis finished:
     similarity is a matrix with 118 rows and 118 columns
```

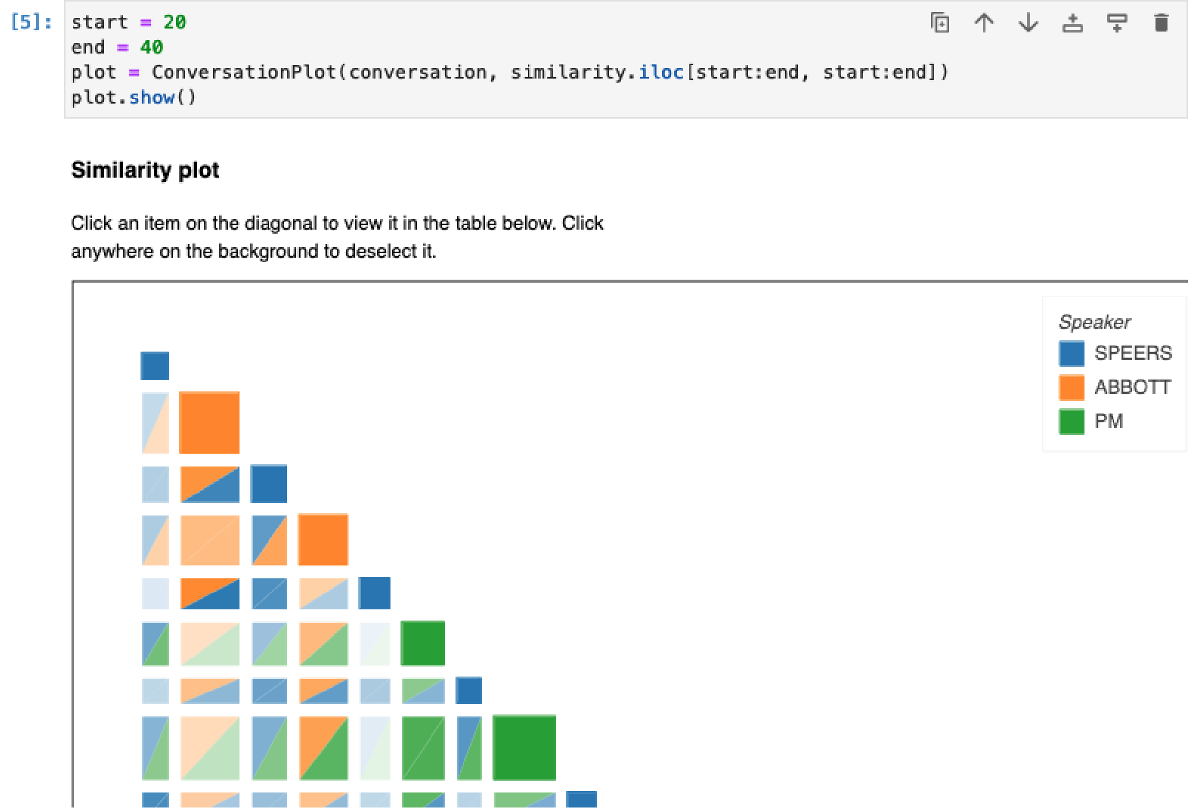
Visualize analysis

Finally, execute the cell under “visualize analysis” to produce a plot visualizing the similarity of topics across the conversation.

You can hover the mouse over tiles on the main diagonal to see the text of each utterance, or hover over tiles below the diagonal to see the similarity scores between each pair of utterances.

You can click on tiles the relevant utterances appear in the table below the plot

To export an image of the plot, select the save icon (💾) at the right of the plot.



```
[5]: start = 20
     end = 40
     plot = ConversationPlot(conversation, similarity.iloc[start:end, start:end])
     plot.show()
```

Similarity plot

Click an item on the diagonal to view it in the table below. Click anywhere on the background to deselect it.