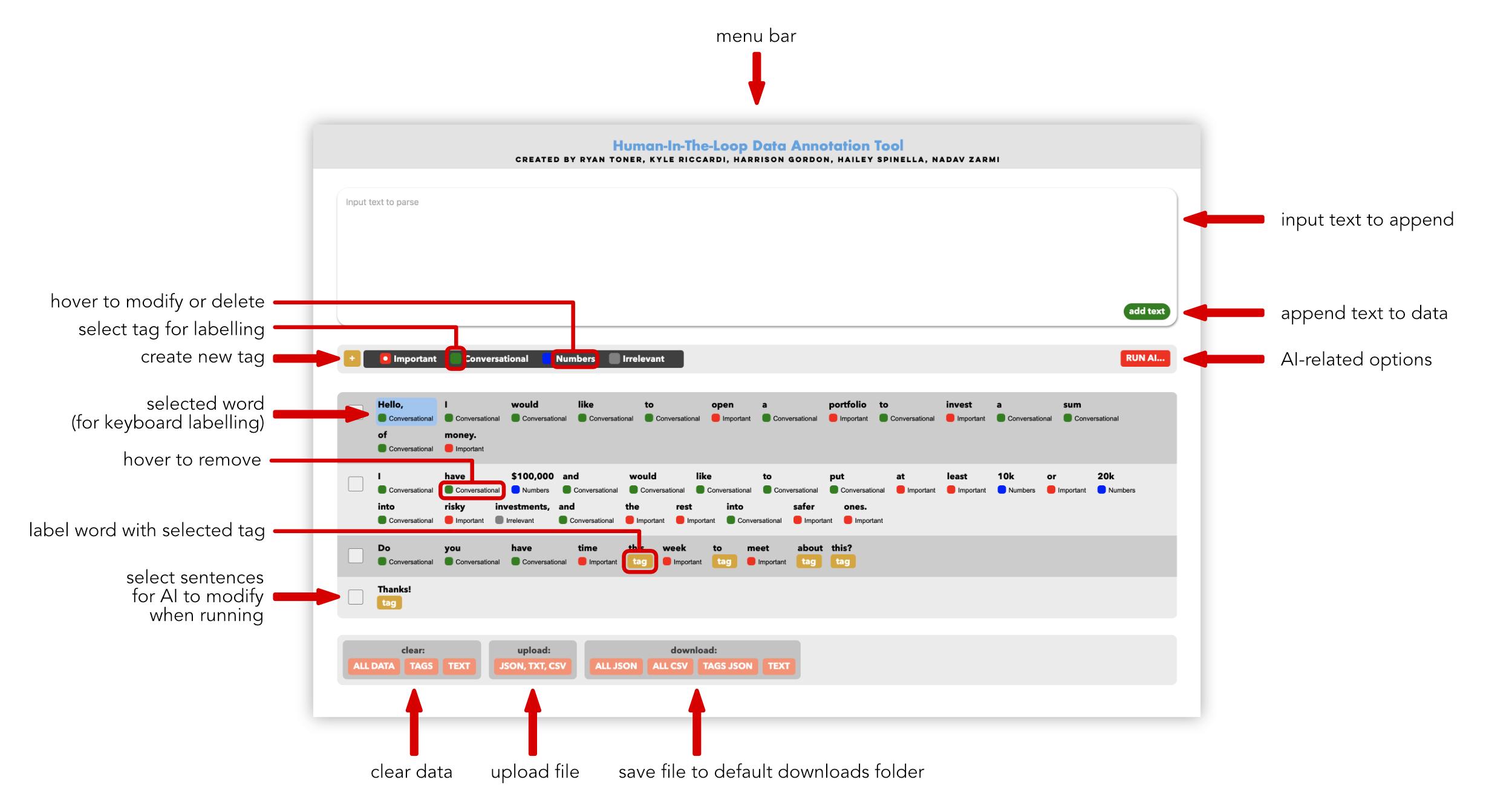
**Data Annotation Tool User Manual**

**Definitions:**

Sentence — one sentence in text, delimited by a period, question mark, or exclamation, followed by a space

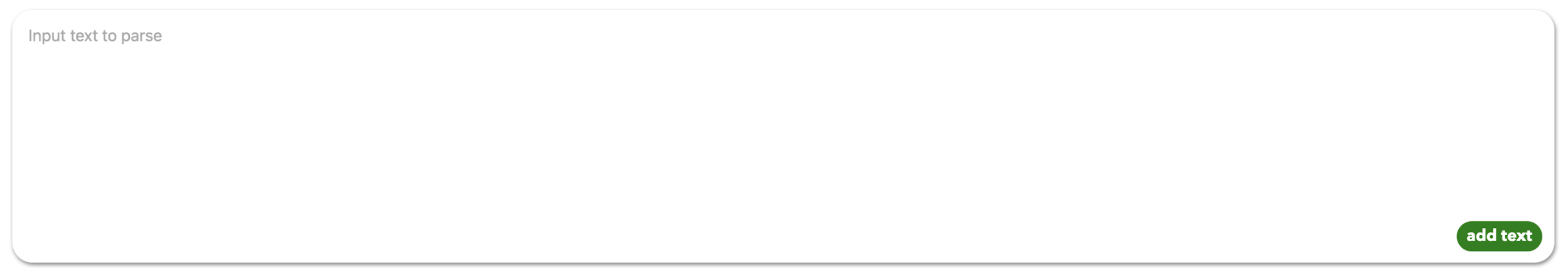
Tag — a pair of a name and color used to label words

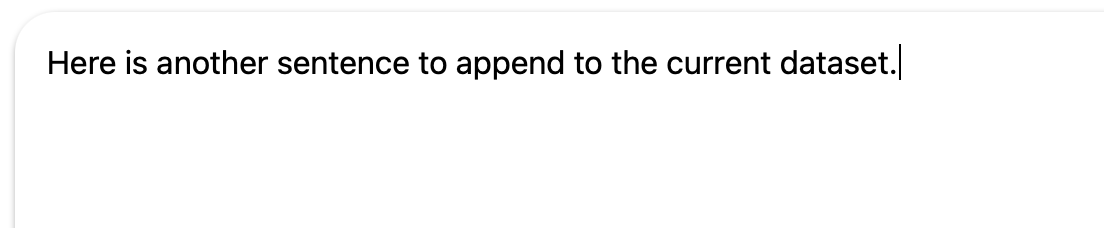
Label — the correlation of a tag to a word, e.g. you label word *x* with tag *y*

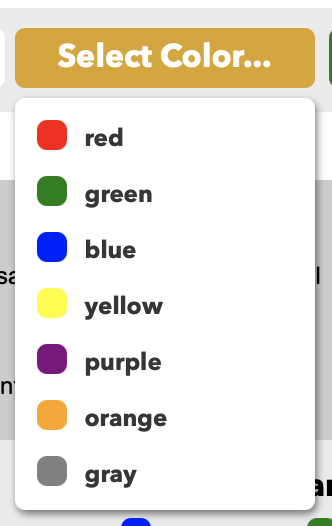
****

**Interface:**

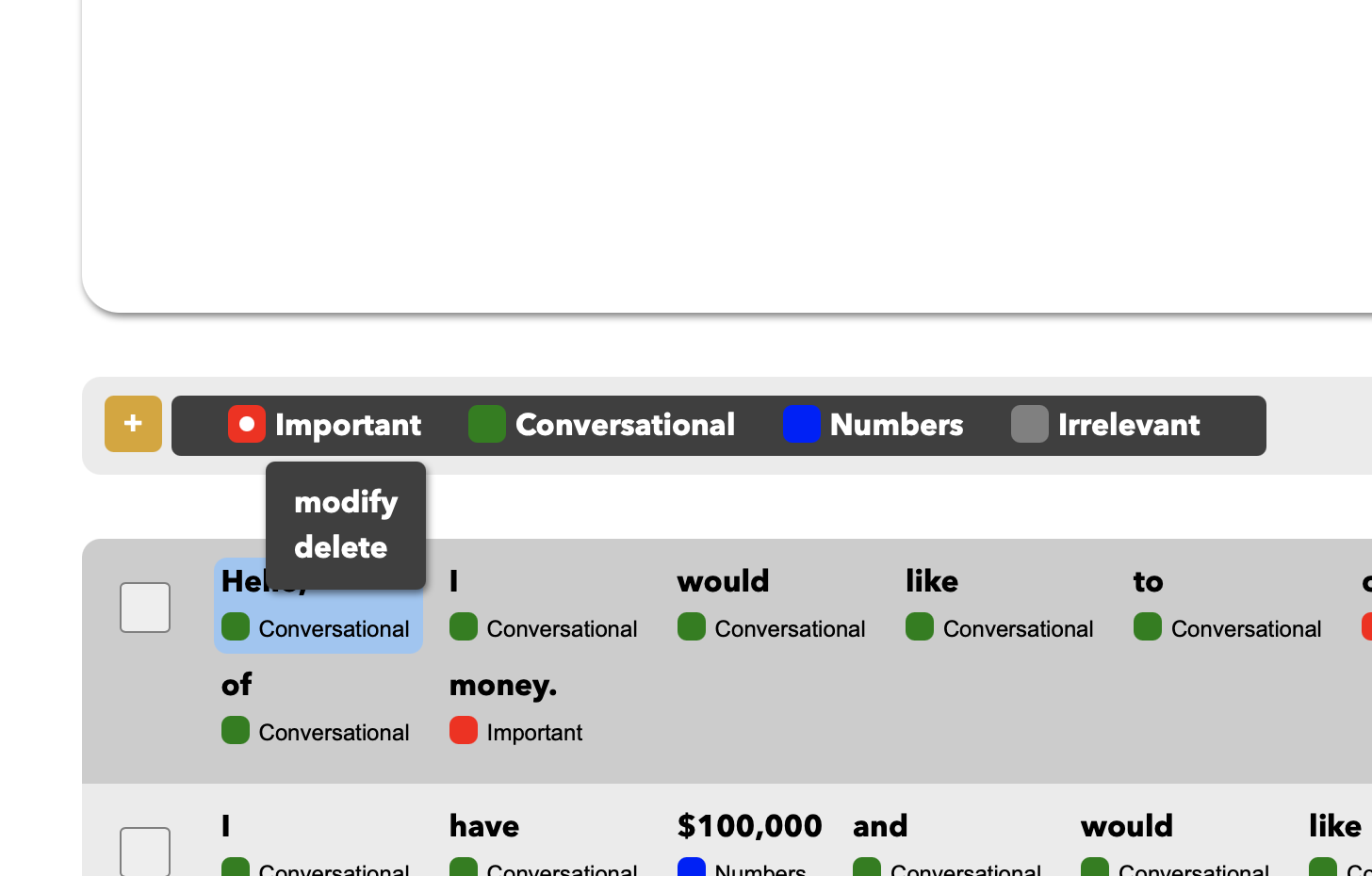
Menu bar — contains logo (also functions as a home button)

Text box — used for typing or pasting text to upload to the dataset. Click add text to split the sentences into the sentence area.



Tags bar — used to create, use, and manage tags

Plus button — create a new tag, must be unique

Name — enter a name for the tag, cannot be empty

Color — select a color for the tag from the dropdown

Done — add the new tag

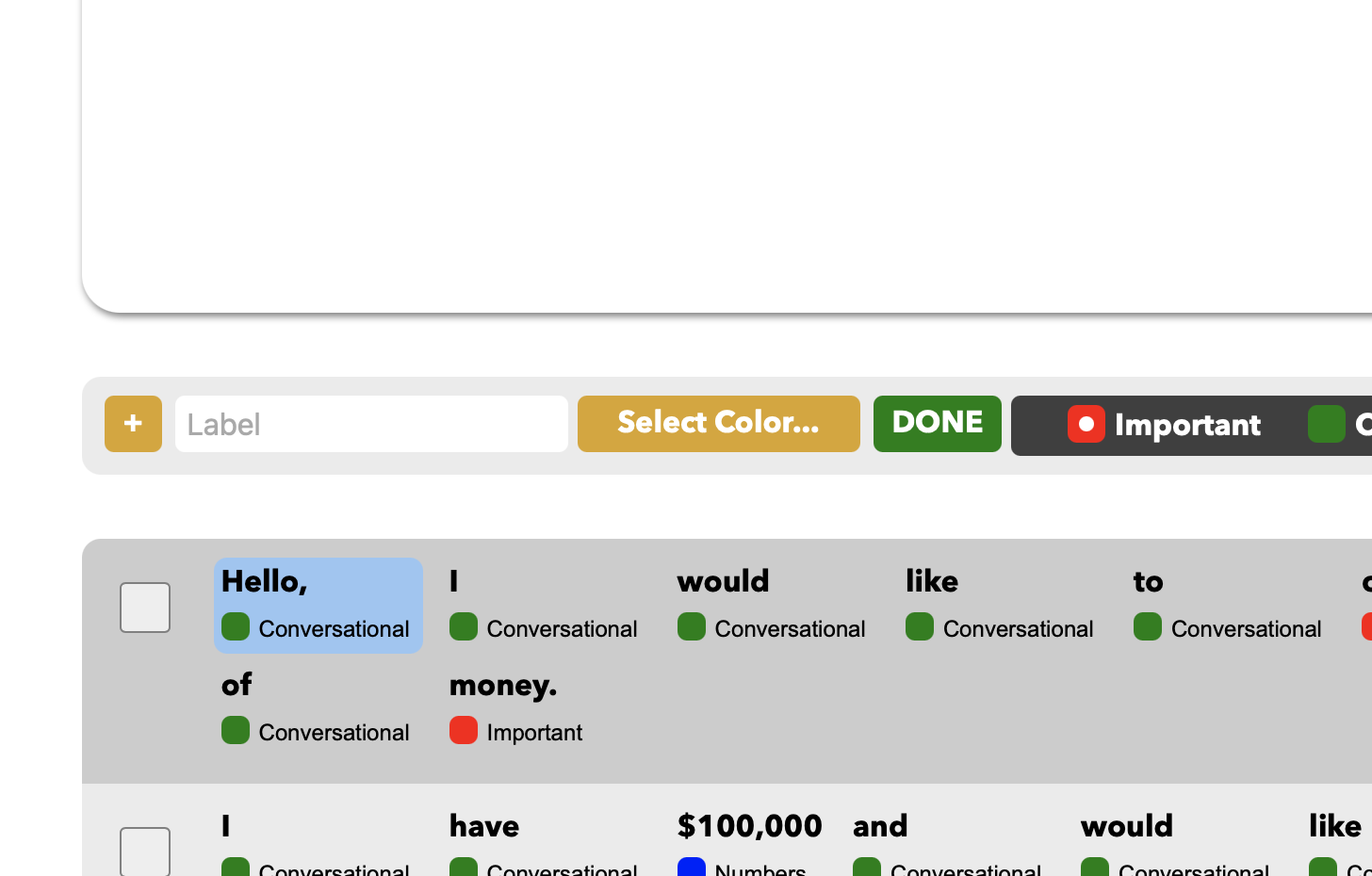
Tags manager — select and manage tags

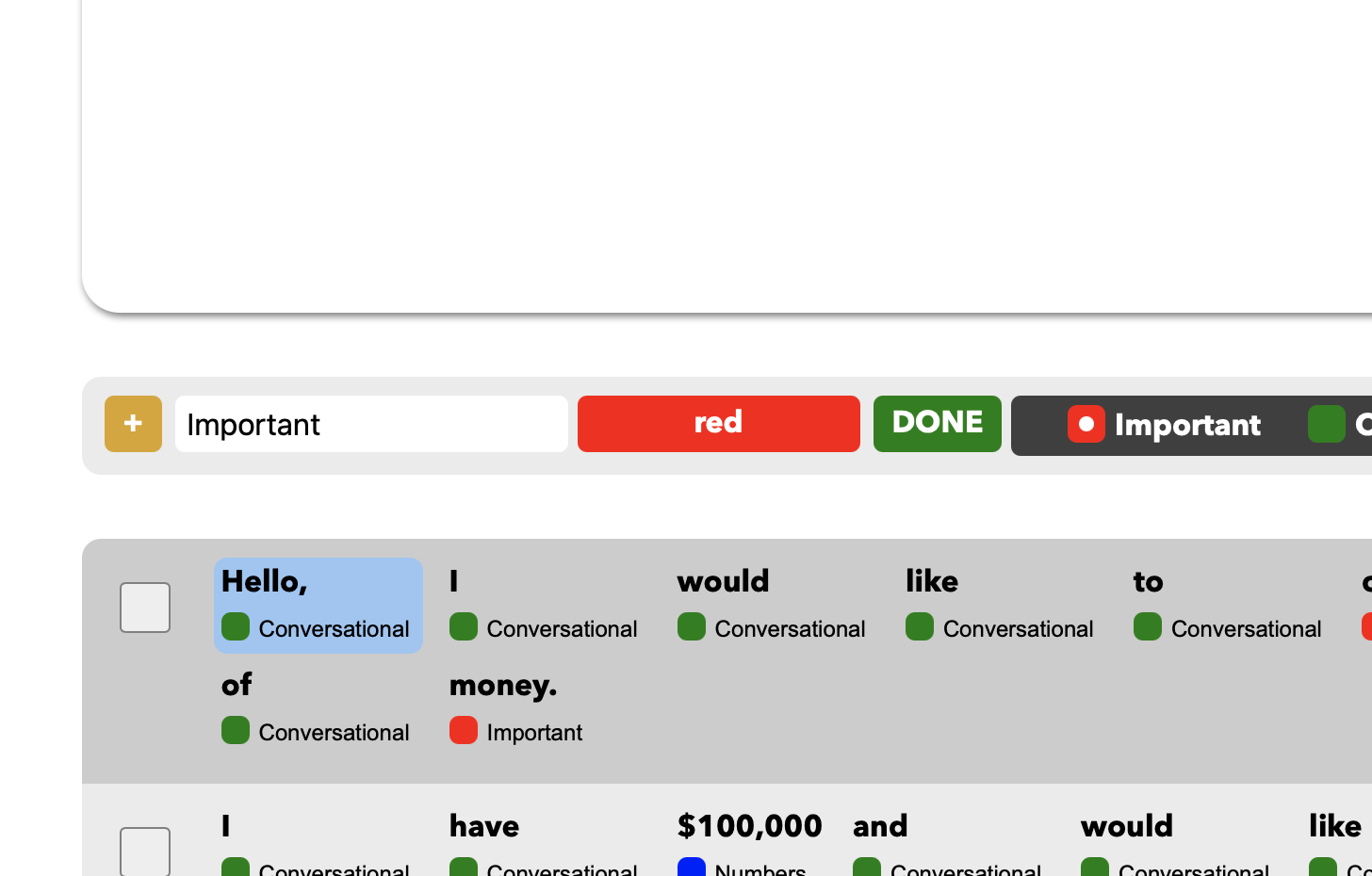
Tag checkbox — makes tag active for labeling

Tag edit — hover over tag to bring up edit menu

Modify — opens menu to change tag name or color

Delete — deletes tag from manager and the entire datas





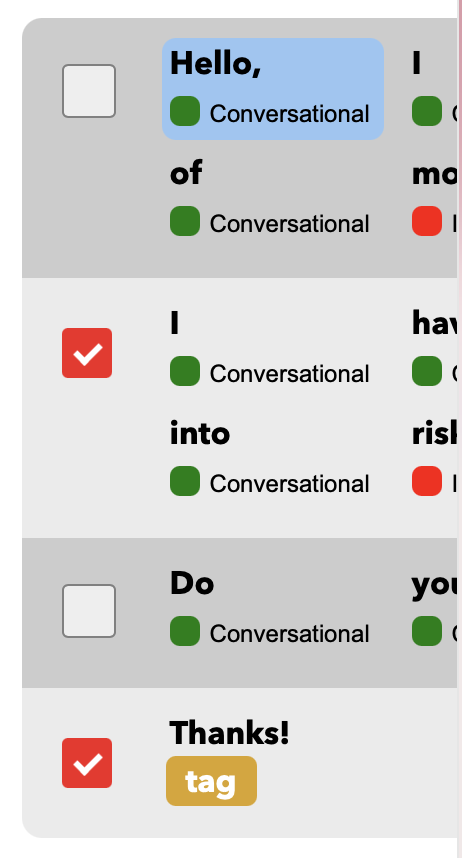
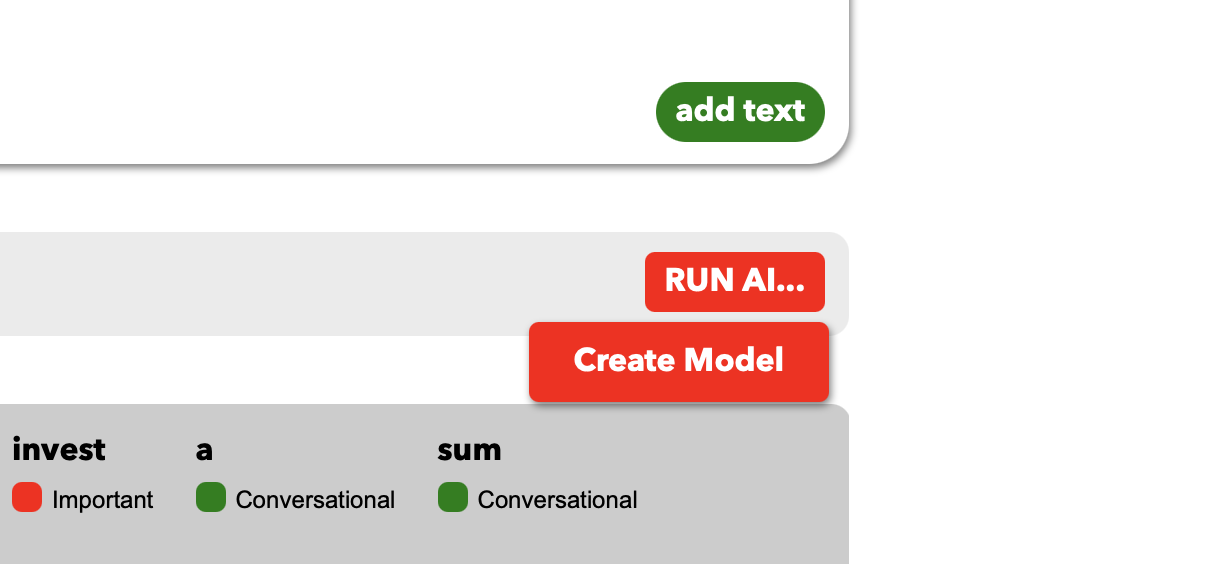
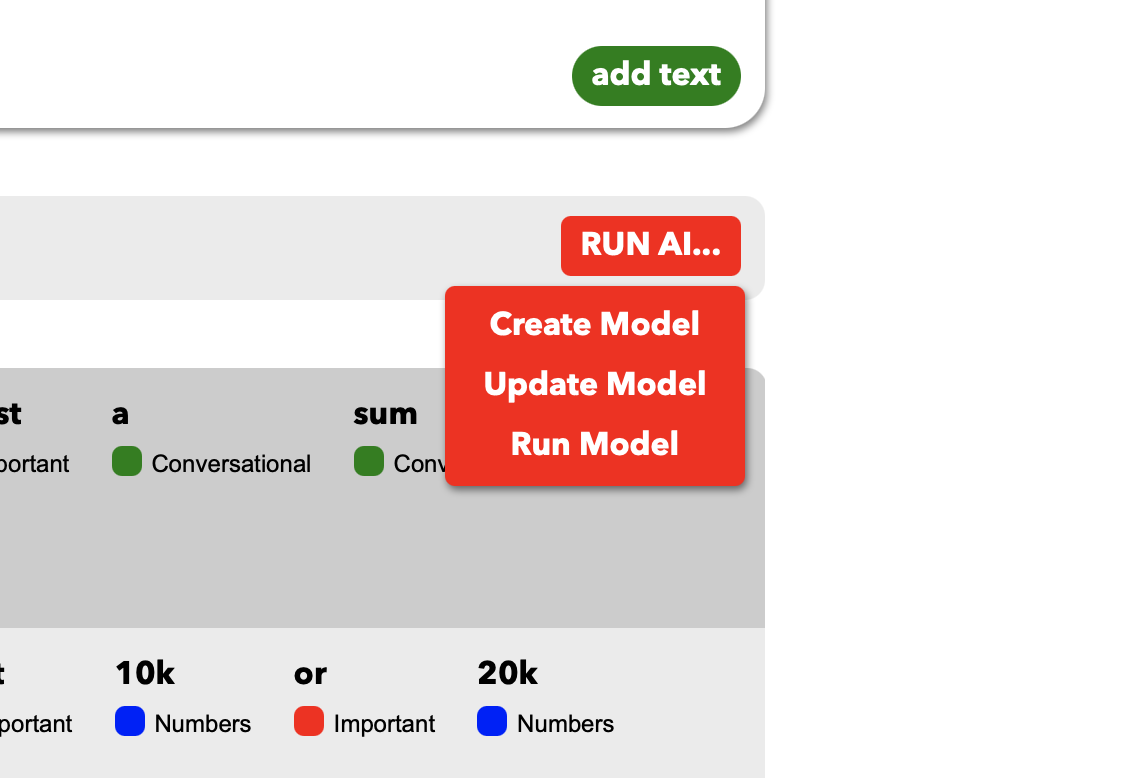
Run menu — used to specify the mode of data processing

Create Model — create new model trained on data

Load Model — load existing model compatible with dataset

Update Model — train existing model on data

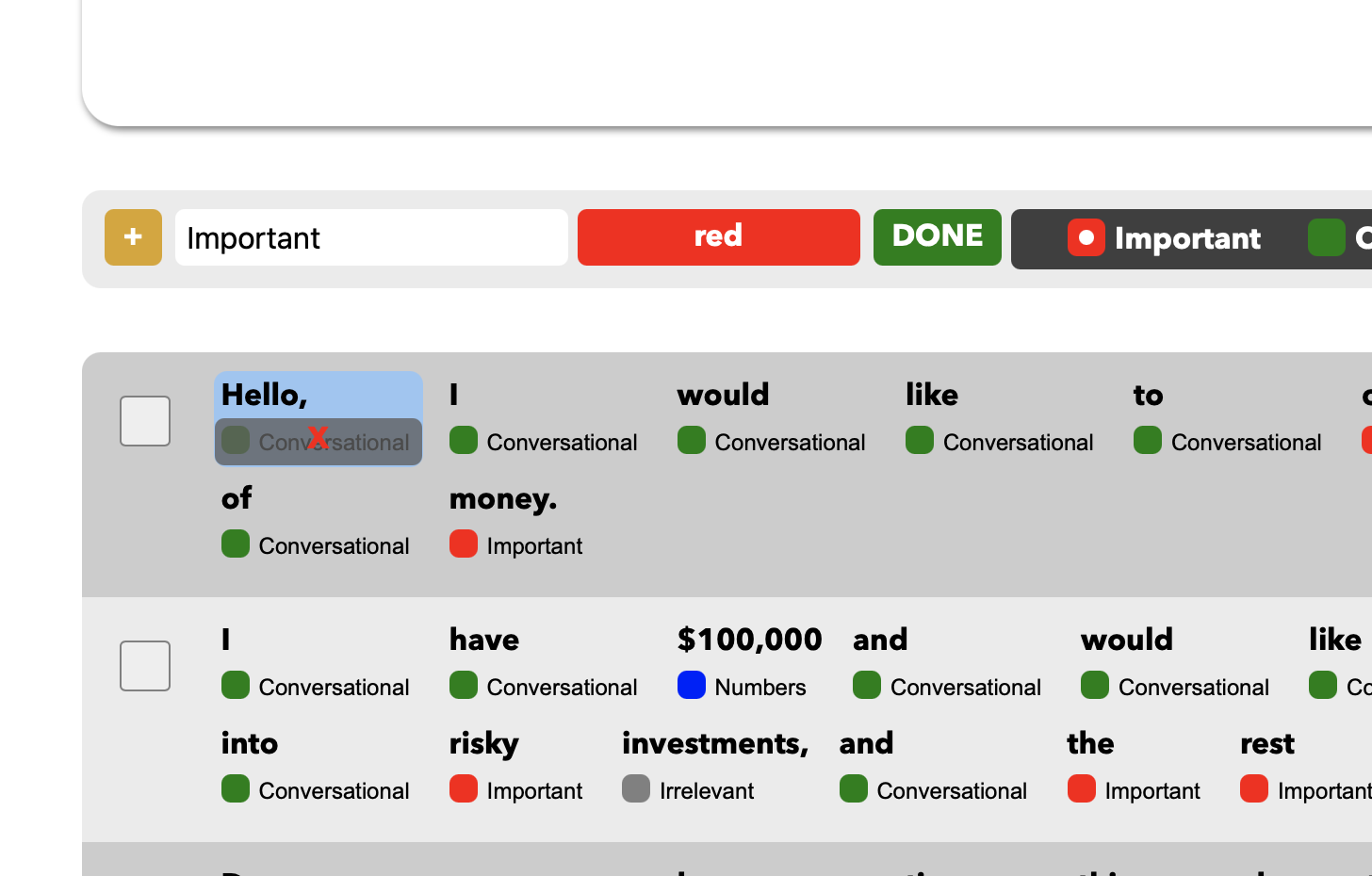
Run Model — use existing model to modify data



Sentences area — where all text and labels are visible

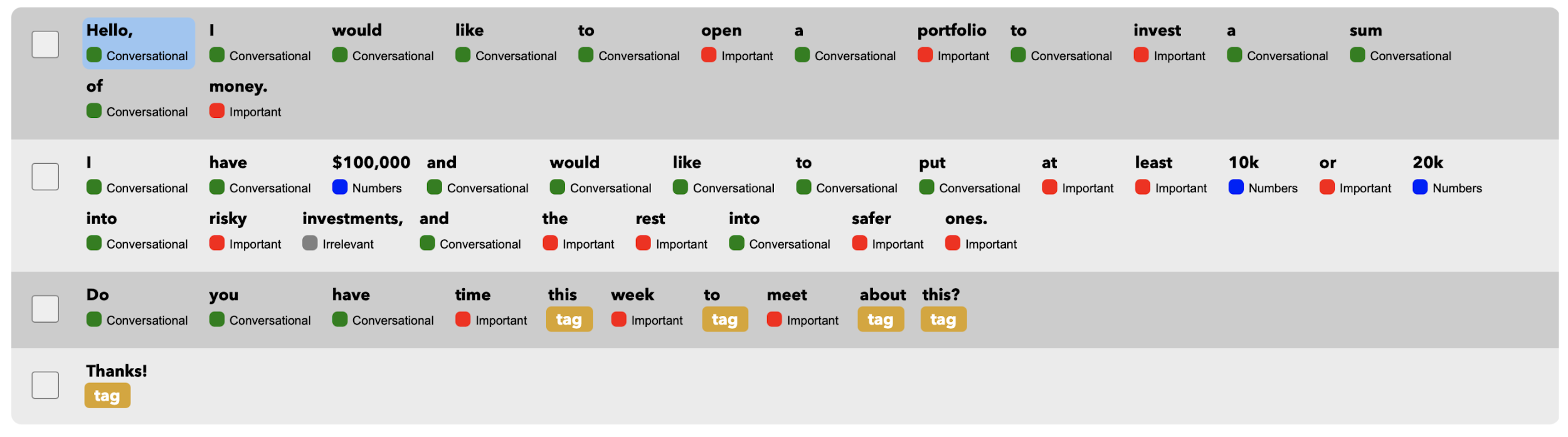
Sentence — sentences are separated for visibility by delimiters [”. ”, “? ”, ”! ”]

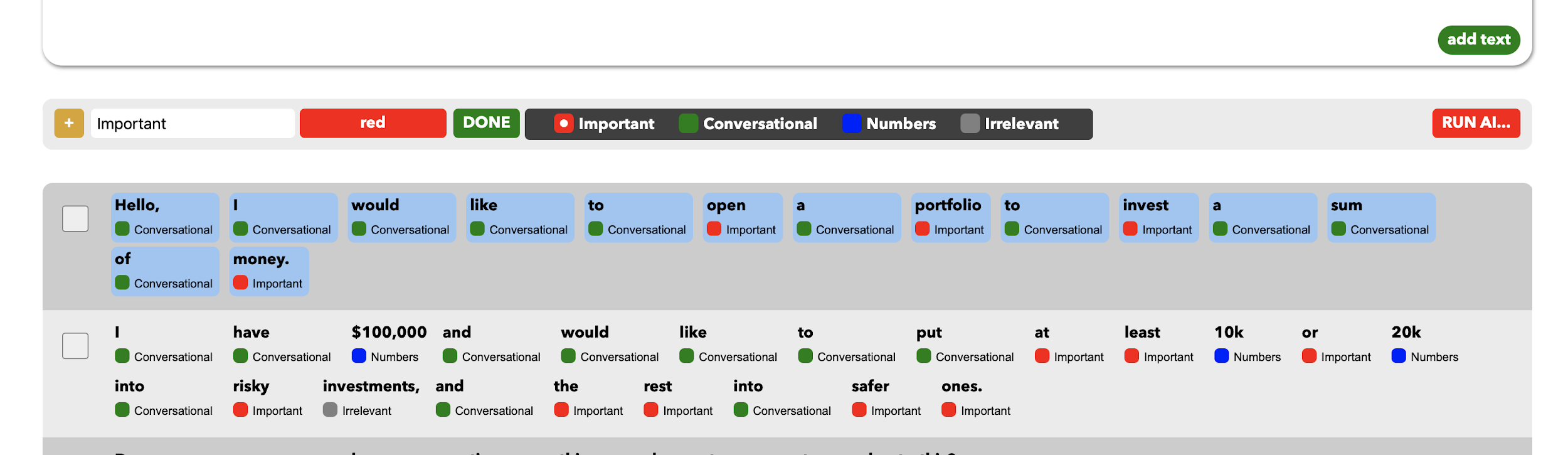
*An entire sentence can be selected for batch label editing using Shift-A*

*Checkboxes — select sentences for modification by AI when running*

Word — may be selected one at a time using keyboard or label changes

Tag button — click the tag button below a word to add

X button — click the X to remove a tag from a word



Footer — includes actions for clearing, uploading, downloading, and saving data

Clear — deletes data from the current dataset

All data — deletes all data from the current dataset (sentences, tags, labels, unsaved model)

Tags — deletes tags and labels (leaving only raw text)

Text — deletes text (and therefore labels, leaving only tags)

Model — removes the current model from memory (does not delete from disk)

Upload — click to upload a JSON, CSV, or TXT file

JSON — upload a JSON file like that created by a save action

All data — upload a new dataset, clearing all data in the

current session; user is prompted to save dataset before

overwriting

Tags only — append tags to the current dataset; nothing is

cleared or removed (no need to save)

CSV — upload a CSV file like that created by an export action

All data — upload a new dataset, clearing all data in the

current session; user is prompted to save dataset before

overwriting

TXT — upload a TXT file of any raw text, appending to the current dataset (no need to save)

Download — choose to download your data through the web browser

All JSON — saves sentences, tags, and labels in a JSON file

All CSV — saves all data in a CSV, in a more readable format

Tags JSON — saves only the tags (not the labels) in a JSON file

Text — saves a TXT of all sentences as raw text (no labels)

Save Model — choose to save the model

Start — saves a loaded model to the disk from memory (will not be deleted except by user)





Popup Menus — X button cancels or closes, Okay/Continue button submits or closes (pressing the enter key will trigger this action as well)

**Initial Setup**

1. Clone the Git repository to local machine
2. Set up an environment (virtual works best)
3. Find the path to the requirements file
4. Run ‘pip install -r requirements.txt’
5. Run ‘flask run’ (if you get an error message asking to install python-dotenv, do so and re-run)
6. Open the server, add data, and you’re ready to go

**Usage:**

* Create tag — click plus button, enter name, select color, click done
* Modify tag — hover over tag name and click edit button, click modify in menu, can rename and change color, then click done
* Delete tag — hover over tag name and click edit button, click delete in menu (removes all labels from data too)
* Add text in text box — type or paste text into text box, then choose a run option
* Add text with upload — upload a TXT file
* Label word:
  + With mouse — click checkbox next to tag to select it, click tag button below word
  + With keyboard — if not selected already, select word with arrow keys (selected = blue), press a number key (1-9) that corresponds to a tag indexed left to right, starting at 1
* Remove label:
  + With mouse — click label below word (X appears)
  + With keyboard — with word selected, press the X or 0 key
* Batch-label an entire sentence — press Shift-A, label any constituent words with the mouse or use the keyboard per the above method (this can also be used to clear labels from a dataset quickly)
* Save tags, not text or labels — click download: tags (can be re-uploaded later)
* Create new dataset and save current one — click download: all, click clear: all
* Create new dataset, save the current one, and reuse the tags:
  + EITHER — click download: tags, click upload and select new dataset, save current dataset when prompted
  + OR — click download: all (CSV or JSON), click clear: text, and add new text by uploading, or typing or pasting in text box
* Switch to another dataset and save current one — click upload and select new dataset, save current dataset when prompted
* Switch to another dataset, save the current one, and reuse the tags — click download: tags, click upload and select new dataset, save current dataset when prompted, click upload and select the tags that were just saved
* Export a simple CSV file of all the data — click download: CSV (this can be uploaded later to restore application state in its entirety)
* Export raw text and start working on blank slate — click download: text, click clear: all
* Create new AI model — click run AI, click create model, enter a name and continue
* Load existing model — click run AI, click load model, choose one from the dropdown
* Update existing model with annotation corrections — click run AI, click update model
* Run the current model — use the checkboxes to select sentences to be modified, click run AI, click run model
* Remove a loaded model from memory, or clear an unloaded model’s reference in the dataset (and allow tags to be edited again)

**Explanation and Notes:**

* Files, saving, and management:
  + In the working folder, there is a directory called data — this is the only place that files are modified, created, or deleted by the application
    - Within this directory, may be a data.json file, an upload folder, and an ai folder (although the app could need to create them first)
    - The data.json file is where all working data is stored
    - The upload folder is where uploaded files are temporarily stored
    - The ai folder is where saved models are stored
  + The application works off of a continuously-updated JSON file
    - Therefore, any changes made to any data within the app (other than the model) are immediately saved, or transferred from memory to the disk
    - The user has the ability to download this file through their web browser, and it can be re-uploaded to restore the dataset if it is cleared or modified
    - The JSON file is proprietary, which is why the app also offers a legible CSV format in the download options. It is completely interchangeable with the JSON, simply requiring more processing power to create
    - Here’s the exact file structure with explanations of each variable:
    - # JSON file —> user\_data
    - # one dictionary: "sentences", "sentence\_tags", "tag\_data"
    - # user\_data["sentences"] = a list of strings, each string is one sentence fully intact
    - # user\_data["sentence\_tags"] = a list of lists of integers, each child list represents a sentence, each number represents a word's tag index number
    - # the sentences and sentence\_tags data can be correlated by list[i]
    - # user\_data["tag\_data"] = a dictionary: "index", "tags"
    - # user\_data["tag\_data"]["index"] = running index (integer) of how many tags have ever been created, so each new tag has a unique index ID number
    - # user\_data["tag\_data"]["tags"] = a dictionary: keys = tag ID numbers (strings), values = dictionary: "name", "color"
    - # user\_data["tag\_data"]["tags"][tag ID number]["name"] = name of tag (string)
    - # user\_data["tag\_data"]["tags"][tag ID number]["color"] = color of tag (string)
    - # user\_data["model\_name"] = name of currently loaded model, if any (key may not exist)
    - Note, the download: tags option creates a file identical to the user\_data JSON seen above, containing only the “tag\_data” key
  + The option to download and upload a CSV is for easy ex-application data manipulation and viewing
    - It’s format is simple: each row is for a word in the data, inclusive of punctuation
    - The columns are as such: word, tag name, tag color, tag index
      * Preserving the tag index numbers on import allows for a previous AI model to be used
  + The AI model:
    - When a model is created or loaded, it is put into memory and its name is saved to the JSON file
    - The model itself is not automatically saved — the user must click the save model button
    - Only models trained on a set of tags identical to the current dataset (inclusive of indices) will be available to load
    - When saved, it will be kept in a folder in the data/ai directory titled by the model name given upon creation
    - During saving, the current set of tags will be copied into the folder to ensure that it can only be loaded into datasets with that exact tag configuration