Our goal was to train a network to “read” a document containing information about characters and their relationships and answer a question about one or more of those characters.

To collect the necessary data, we scraped summaries of literary works from two electronic encyclopedias accessible through the Harvard library: Master Plots, Editions II and IV.

In addition, we scraped a separate encyclopedia that focused specifically on descriptions of characters.

We converted these descriptions into sets of “queries” that could be answered by reading the summary of the associated work.

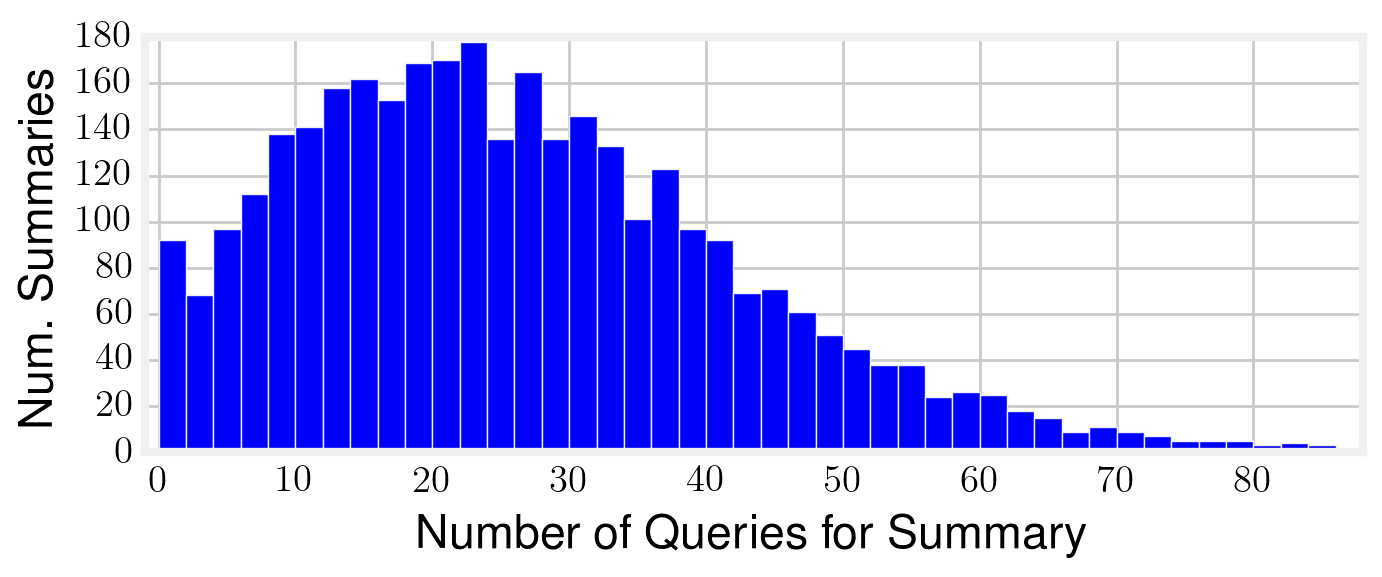
Therefore, a single training example had the form of:

Network input[**Summary document** + **Query**] + [**Answer**]Target Output

For example:

|  |  |  |
| --- | --- | --- |
| Document | Query | Answer |
| Summary of *Romeo and Juliet* | “\_\_\_\_\_ loves Juliet” | “Romeo” |
| Summary of *Romeo and Juliet* | “Romeo loves \_\_\_\_\_” | “Juliet” |
| Summary of *The Lion King* | “\_\_\_\_\_ can’t wait to be king” | “Simba” |

We collected **3314** summary documents that could be paired with character descriptions. Because individual works contain many details about characters and their relationships, many queries can be asked of a single summary document, allowing us to extract a total of **86,432** unique document/query pairs.



To avoid teaching the network a degenerate solution, each character was anonymized and the anonymous identities were permuted before each training step (DEEPMIND CITATION). For instance, here is how 3 permutations of a single training example might look:

|  |  |  |
| --- | --- | --- |
| Document | Query | Answer |
| “… ent003, despite circumstance, is in love with ent073 …” | “\_\_\_\_\_ loves ent073” | 3 |
| “… ent021, despite circumstance, is in love with ent002 …” | “\_\_\_\_\_ loves ent002” | 21 |
| “… ent112, despite circumstance, is in love with ent041 …” | “\_\_\_\_\_ loves ent041” | 112 |