Text Splitters

Once you've loaded documents, you'll often want to transform them to better suit your application. The simplest example is you may want to split a long document into smaller chunks that can fit into your model's context window. LangChain has a number of built-in document transformers that make it easy to split, combine, filter, and otherwise manipulate documents.

When you want to deal with long pieces of text, it is necessary to split up that text into chunks. As simple as this sounds, there is a lot of potential complexity here. Ideally, you want to keep the semantically related pieces of text together. What "semantically related" means could depend on the type of text. This notebook showcases several ways to do that.

At a high level, text splitters work as following:

1. Split the text up into small, semantically meaningful chunks (often sentences).
2. Start combining these small chunks into a larger chunk until you reach a certain size (as measured by some function).

3. Once you reach that size, make that chunk its own piece of text and then start creating a new chunk of text with some overlap (to keep context between chunks).

That means there are two different axes along which you can customize your text splitter:

1. How the text is split
2. How the chunk size is measured

Types of Text Splitters

LangChain offers many different types of text splitters. Below is a table listing all of them, along with a few characteristics:

Name: Name of the text splitter

Splits On: How this text splitter splits text

Adds Metadata: Whether or not this text splitter adds metadata about where each chunk came from.

Description: Description of the splitter, including recommendation on when to use it.

Name	Splits On	Adds Metadata	Description
Recursive	A list of user defined characters		Recursively splits text. Splitting text recursively serves the purpose of trying to keep related pieces of text next to each other. This is the recommended way to start splitting text.
HTML	HTML specific characters	✅	Splits text based on HTML-specific characters. Notably, this adds in relevant information

Name	Splits On	Adds Metadata	Description
			about where that chunk came from (based on the HTML)
Markdown	Markdown specific characters	✅	Splits text based on Markdown-specific characters. Notably, this adds in relevant information about where that chunk came from (based on the Markdown)
Code	Code (Python,		Splits text based on

Name	Splits On	Adds Metadata	Description
	JS) specific characters		characters specific to coding languages. 15 different languages are available to choose from.
Token	Tokens		Splits text on tokens. There exist a few different ways to measure tokens.
Character	A user defined character		Splits text based on a user defined character. One of the simpler methods.

Name	Splits On	Adds Metadata	Description
[Experimental] Semantic Chunker	Sentences		First splits on sentences. Then combines ones next to each other if they are semantically similar enough. Taken from Greg Kamradt

Evaluate text splitters

You can evaluate text splitters with the Chunkviz utility created by `Greg Kamradt`. `Chunkviz` is a great tool for visualizing how your text splitter is working. It will show you how your text is being split up and help in tuning up the splitting parameters.

Other Document Transforms

Text splitting is only one example of transformations that you may want to do on documents before passing them to an LLM. Head to Integrations for documentation on built-in document transformer integrations with 3rd-party tools.