Notes on the DSPy Model

compiled by D.Gueorguiev, 6/24/25

DSPy framework treats LMs as abstract devices for text generation, and optimizes their usage in arbitrary computation graphs. DSPy programs are written in Python – each program takes a task input (e.g. a question to answer or a paper to summarize) and returns the output (e.g. an answer or a summary) after series of steps. DSPy presents three abstractions toward automatic optimization – *signatures*, *modules,* and *teleprompters*.

*Signatures* abstract the input/output behavior of a module. *Modules* replace existing hand-prompting techniques and can be composed in arbitrary pipelines. *Teleprompters* optimize all modules present in a pipeline to maximize a metric.

Details on *Signatures*

Instead of free form string prompts, DSPy programs use natural language *signatures* to assign work to the LM.

A DSPy signature is a *natural-language typed* declaration of a function: a short declarative spec that tells DSPy **what** a text transformation needs to do (e.g. *“consume questions and return answers”*), rather than **how** a specific LM should be prompted to implement that behavior.