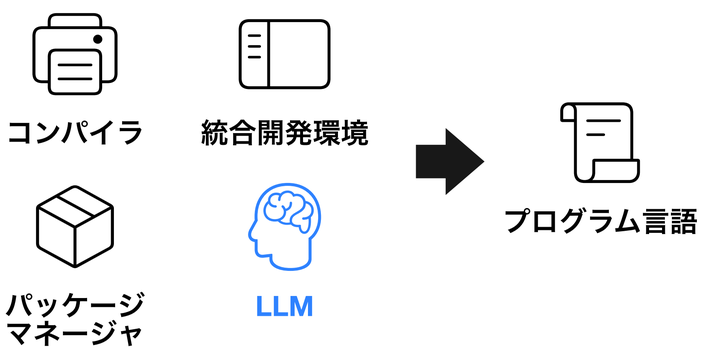
現在存在しないプログラミング言語向け大規模言語モデルの開発に向けた予備実験

大渕雄生 (東京大学)・千葉滋 (東京大学)

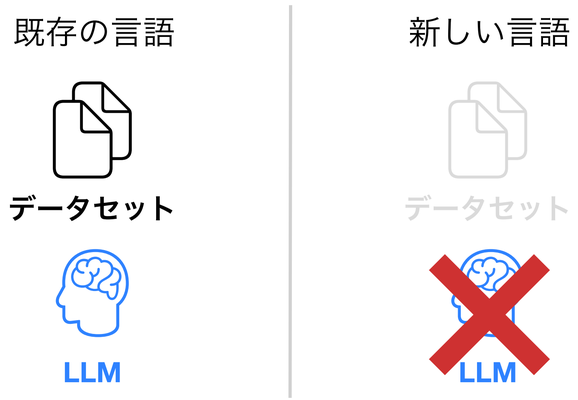
新プログラミング言語におけるデータ不足はLLMの適応を阻み、新言語の普及を阻む、この問題を克服する必要がある。

プログラム言語の普及には様々な周辺ツールが必要



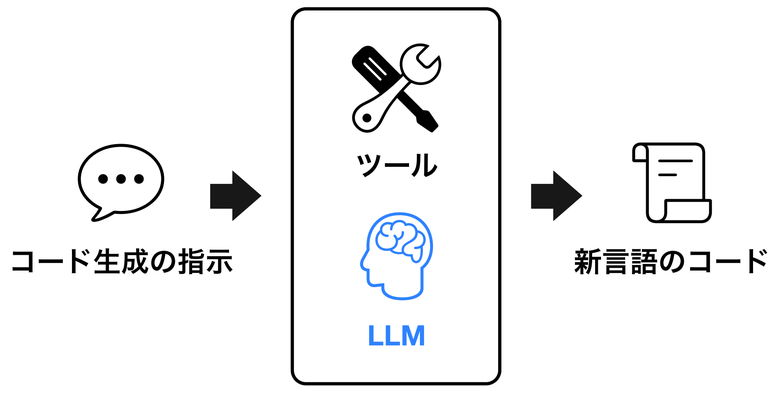
これらのツールチェーンに
LLMが新しいツールとして加わりつつある

新しいプログラミング言語にはデータセットがない



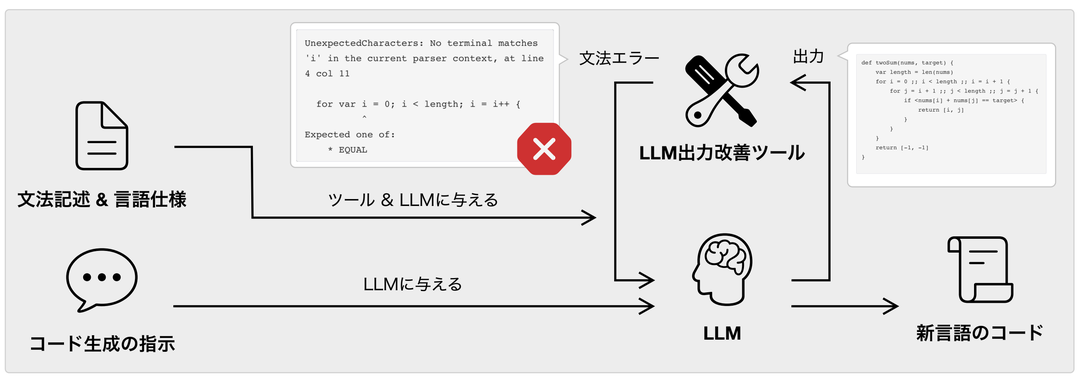
データセットがないために
LLMツールを作成することができない

データセットなしで新言語に対応するツールを作る



言語開発者が大量のデータを作成しなくとも、LLMを新言語に対応させるツールを作る

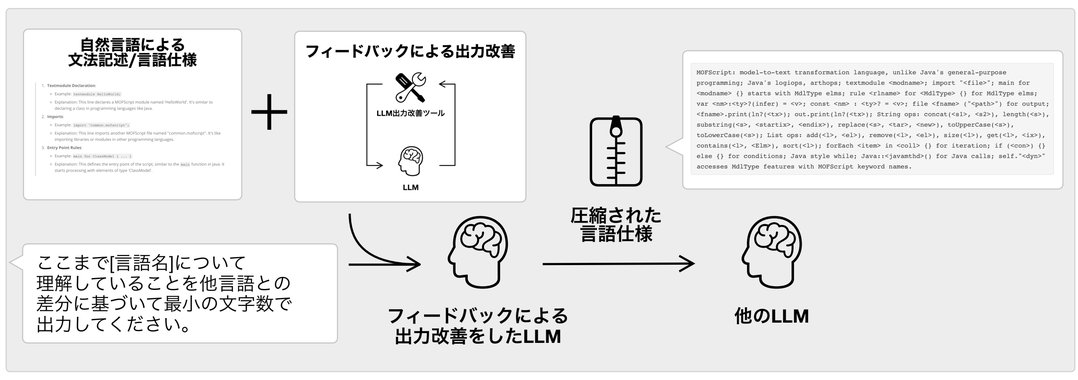
手法1. エラーフィードバックによる出力改善



LLMにコード生成の課題を与え、その出力に対して、LLM出力改善ツールが文法エラーをフィードバックする。これを文法的エラーがなくなるか、指定の回数まで繰り返すことでLLMの出力を改善する。

LLMには自然言語で記述した新しいプログラミング言語の文法をプロンプトとして与える。LLM出力改善ツールにはBNFで記述した形式的な文法記述を与える。

手法2. 言語仕様の圧縮によるトークン数の節約



「エラーフィードバック」による出力の改善を行なったLLMに、既存言語との差異などを中心に、他のLLMに伝達可能なように最小限のトークン数でコンテキストを圧縮するように命じる。

これによって、トークン数を節約し、LLMに規模の大きな文法をプロンプトで渡せるようにする。

予備実験の結果

LLMにとって未知な3つの言語（PrintTalk・MofScript・Simplicity）を用いて、アイディアの実現可能性を検証するため予備実験を行った。

データセット：論文のコード例、公式ドキュメントのコード例、類似言語の再現、インターネット上のコード例及び、その改変
評価基準：生成されたプログラムが構文エラーを含むかどうか

エラーフィードバックによる効果

	PrintTalk	MofScript	Simplicity
出力改善前の正答率	50% (3/6)	28% (2/7)	44% (4/9)
出力改善後の正答率	**83% (5/6)**	**71% (5/7)**	**66% (6/9)**

これらの言語を用いて「エラーフィードバックによる出力改善」を行なった結果、左表のように改善後の方が課題に対して構文エラーのないプログラムを生成できる率は向上した。

ここで用いた各言語の課題は1つの構文を生成するもの（2件程度）複数の文を持つもの（4件程度）複数の構文を組み合わせる発展的なもの（2件程度 PrintTalkでは未実施）からなる。

言語仕様の圧縮による効果 (MofScriptの例)

	PrintTalk	MofScript	Simplicity
圧縮前のトークン数	未実施	2324	1895
圧縮後のトークン数	未実施	**278**	**258**
圧縮率		**88%**	**86%**

自然言語による文法を与えた上で出力改善を行ったLLMに言語仕様を圧縮するように命じた結果、
左表のようにトークン数はどちらも15%以下になった。
また、圧縮された文法を与えた場合でも実験を実施した課題では自然言語による文法と言語仕様、エラーフィードバックを行なったLLMで正解した課題は全て正解できた。

ここで、MofScriptではエラーフィードバックで正答できた課題から4件、Simplicityでは3件実施した。