UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Curso de Bacharelado em Ciência da Computação

Trabalho de Conclusão de Curso

LLM-Powered Applications: Tecnologia, Questões e estudo de caso.

Marilton Sanchotene de Aguiar

Pelotas, 2024

Marilton Sanchotene de Aguiar

LLM-Powered Applications: Tecnologia, Questões e estudo de caso.

Trabalho de Conclusão de Curso apresentado ao Centro de Desenvolvimento Tecnológico da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Marilton Sanchotene de Aguiar
Coorientador: Prof. Dr. Marilton Sanchotene de Aguiar
Colaborador: Prof. Dr. Marilton Sanchotene de Aguiar

Pelotas, 2024

Insira AQUI a ficha catalográfica
Quando finalizado o trabalho, deve ser solicitada através do Sistema Cobalto
Biblioteca – Cadastro – Ficha catalográfica.

Marilton Sanchotene de Aguiar

LLM-Powered Applications: Tecnologia, Questões e estudo de caso.

Trabalho de Conclusão de Curso aprovado, como requisito parcial, para obtenção do grau de Bacharel em Ciência da Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas.

Data da Defesa: 30 de fevereiro de 2019

Banca Examinadora:

Prof. Dr. Marilton Sanchotene de Aguiar (orientador)
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Paulo Roberto Ferreira Jr.
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Ricardo Matsumura Araujo
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Luciano da Silva Pinto
Doutor em Biotecnologia pela Universidade Federal de Pelotas.

Dedico. . .

AGRADECIMENTOS

Agradeço. . .

Só sei que nada sei.
— SÓCRATES

RESUMO

AGUIAR, Marilton Sanchotene de. **LLM-Powered Applications: Tecnologia, Questões e estudo de caso.**. Orientador: Marilton Sanchotene de Aguiar. 2024. 30 f. Trabalho de Conclusão de Curso (Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2024.

Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla.

Palavras-chave: palavrachave-um; palavrachave-dois; palavrachave-tres; palavrachave-quatro.

RESUMO

AGUIAR, Marilton Sanchotene de. **Titulo do Trabalho em Ingles**. Orientador: Marilton Sanchotene de Aguiar. 2024. 30 f. Trabalho de Conclusão de Curso (Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2024.

Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla.

Palavras-chave: keyword-one; keyword-two; keyword-three; keyword-four.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
NUMA	Non-Uniform Memory Access
SIMD	Single Instruction Multiple Data
SMP	Symmetric Multi-Processor
SPMD	Single Program Multiple Data

SUMÁRIO

1 INTRODUÇÃO

1.1 Grandes Modelos de Linguagem

Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla.

> Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla Moore (1979); Aguiar; Marilton (2005).

Bla blabla blablabla bla. Bla blabla blablabla bla.

Bla blabla blablabla bla. Bla blabla blablabla bla (Neumann; Aguiar, 1966).

1.2 Outra seção

Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla

Tabela 1 – Nome da Tabela

Blabla	Blabla	Blablabla
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla.*
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla.*
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla.*
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla.*
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla.*
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla. Conforme a figura 1*

Fonte: Elaborada pelo autor.

blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla **??**.

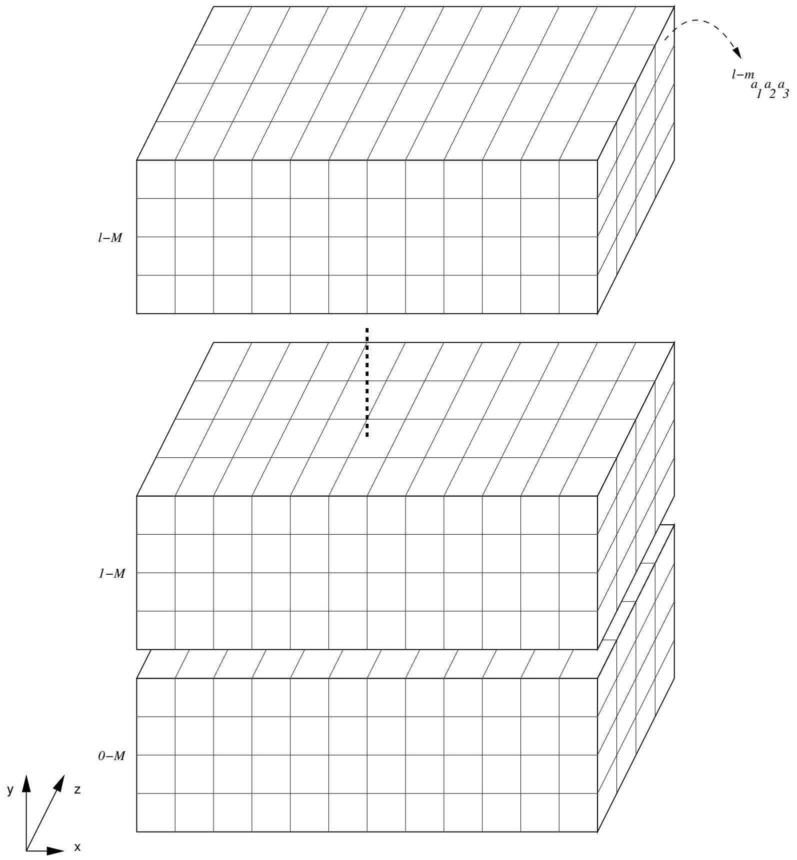
1.2.1 Uma subseção

Bla blabla blablabla bla. Bla blabla blablabla bla.

2 LLM POWERED APPLICATIONS

Bla blabla blablabla bla. Bla blabla blablabla bla 2.

Figura 1 – Nome da figura



Fonte: Elaborada pelo autor.

Tabela 2 – Nome da Tabela

Blabla	Blabla	Blablabla
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla.*
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla.*
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla.*
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla.*
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla.*
Bla	Blabla	*Bla blabla blablabla blabla blablabla blabla blablabla. Conforme a figura 1*

2.1 Fine tuning

2.2 Retrieval Augmented Generation

2.3 Engenharia de prompt

A engenharia de prompt é uma disciplina emergente que desempenha um papel fundamental na utilização e eficiência de modelos de linguagem grandes (LLMs), especialmente dentro do domínio das tarefas de processamento de linguagem natural. A medida que a IA continua a permear diversos setores, a habilidade de comunicar efetivamente com esses sistemas torna-se crucial. Este capítulo tem como objetivo dissecar o conceito de engenharia de prompt, suas técnicas, aplicações e o profundo impacto que ela tem em diferentes campos.

Técnicas de Engenharia de Prompt

Essencialmente, a engenharia de prompt é a prática de formular e refinar prompts para maximizar o desempenho de modelos de linguagem. Um prompt bem elaborado pode alterar significativamente a saída de uma IA, fazendo a diferença entre uma resposta útil e uma irrelevante.

A engenharia de prompt eficaz depende de vários princípios centrais que garantem clareza, especificidade e relevância na interação com sistemas de IA. Esses princípios incluem escrever instruções claras e descritivas, usar delimitadores, fornecer exemplos, atribuir papéis, adicionar informações de contexto, dividir tarefas complexas e solicitar múltiplas soluções. Técnicas avançadas também envolvem a sinergia de raciocínio e ação em modelos de linguagem, como modelos auxiliados por programas, raciocínio automático e uso de ferramentas, e ajuste de prompt para adaptar as

respostas do modelo a necessidades específicas (DaveAI).

Técnicas Avançadas de Prompt

As técnicas avançadas de prompt englobam uma variedade de estratégias projetadas para obter respostas mais sofisticadas de LLMs. Estas incluem prompt de encadeamento de pensamento, que encoraja os modelos a exibir seu processo de raciocínio, e prompt de menor para maior, que guia os modelos por meio de uma série progressiva de complexidade em suas respostas. Além disso, o prompt de geração de conhecimento e o uso de modelos de árvore de pensamento representam abordagens inovadoras para a resolução de problemas com IA (Arxiv).

Aplicações da Engenharia de Prompt

As aplicações da engenharia de prompt são vastas e variadas. Na escrita acadêmica, os pesquisadores podem aproveitar a engenharia de prompt para simplificar revisões de literatura, sintetizar informações complexas e até mesmo gerar rascunhos de artigos. Além da academia, a engenharia de prompt é instrumental em áreas como atendimento ao cliente, onde a IA pode fornecer respostas personalizadas a perguntas, e em indústrias criativas, onde a IA pode auxiliar na criação de conteúdo que varia da escrita à geração de imagens (Intellyverse).

Impacto da Engenharia de Prompt

O impacto da engenharia de prompt é substancial, pois influencia diretamente a eficácia das interações de IA. Ao otimizar prompts, os usuários podem obter saídas mais precisas e contextualmente relevantes dos sistemas de IA, aumentando assim a produtividade e reduzindo o potencial de mal-entendidos. O campo também levanta importantes considerações sobre o uso ético da IA, já que os prompts em si podem moldar a natureza das informações fornecidas pela IA (Giray).

Conclusão

A engenharia de prompt não é apenas uma habilidade técnica, mas uma forma de arte que requer um profundo entendimento tanto da linguagem quanto da tecnologia. À medida que a IA se torna cada vez mais sofisticada, o papel do engenheiro de prompt se tornará ainda mais crítico na moldagem das interações entre humanos e máquinas. A disciplina está no cruzamento da comunicação, tecnologia e criatividade, oferecendo uma nova fronteira para exploração e inovação na era da IA.

Lista de Referências

- Giray, Louie. "Engenharia de Prompt com ChatGPT: Um Guia para Escritores Acadêmicos."*Springer Nature*, 2023, https://link.springer.com/article/10.1007/s10439-023-03272-4.
- "Como se Tornar um Engenheiro de Prompt."*DataCamp*, https://www.datacamp.com/blog/how-to-become-a-prompt-engineer.
- "Guia Completo de Engenharia de Prompt."*DaveAI*, https://daveai.substack.com/p/prompt-engineering-full-guide.
- "Sinergizando Raciocínio e Ação em Modelos de Linguagem."*Arxiv*, https://arxiv.org/abs/2210.03629.
- "Engenharia de Prompt."*Intellyverse*, https://intellyverse.com/blog/prompt-engineering.

2.4 DSPy e a Transformação da Engenharia de Instruções em Modelos de Linguagem de Grande Escala

Resumo

O surgimento do DSPy marca um marco significativo na evolução do desenvolvimento de aplicativos de Modelos de Linguagem de Grande Escala (LLMs). Ao introduzir instruções programáticas, o DSPy aborda a fragilidade inerente às aplicações baseadas em LLM e transforma a prática da engenharia de instruções. Este capítulo examina o framework DSPy, suas implicações para a engenharia de instruções e o impacto mais amplo no desenvolvimento de sistemas inteligentes.

Introdução

O campo da inteligência artificial testemunhou o surgimento de LLMs, que se tornaram cruciais em várias aplicações. No entanto, a complexidade e a fragilidade na construção dessas aplicações apresentam desafios que o DSPy busca superar. O DSPy substitui a engenharia de instruções tradicional por uma abordagem centrada na programação, oferecendo aos desenvolvedores um método robusto e sistemático para instruir LLMs (Monigatti). Este capítulo adentra o framework do DSPy e seu efeito transformador no desenvolvimento de aplicações GenAI.

Framework DSPy

O DSPy, desenvolvido por pesquisadores como Matei Zaharia, oferece uma abordagem inovadora para o desenvolvimento de aplicações LLM. Ele desloca o foco da elaboração de instruções específicas para um modelo de programação que encapsula todo o pipeline de interações LLM. O framework fornece módulos componíveis

e declarativos com uma sintaxe Pythonica, permitindo aos desenvolvedores escrever código livremente com construções familiares como loops, instruções if e exceções (Datanami).

Instrução Programática

No cerne do DSPy está o conceito de instrução programática, que permite aos desenvolvedores definir o comportamento dos LLMs através de código em vez de instruções ad hoc. O framework inclui um compilador que otimiza automaticamente as etapas declarativas do programa, abstraindo assim a complexidade da engenharia de instruções. Este método não só simplifica o processo de desenvolvimento, mas também melhora a robustez e a escalabilidade de sistemas baseados em LLM (Stanford NLP).

Otimização de Desempenho

O DSPy demonstrou sua eficácia através de melhorias significativas em métricas de desempenho. Por exemplo, em pipelines autoiniciados, os programas compilados do DSPy superaram as instruções padrão de poucas etapas e as demonstrações criadas por especialistas, alcançando ganhos de mais de 25% e 65% para GPT-3.5 e llama2-13b-chat, respectivamente (arXiv). Esses resultados destacam o potencial do DSPy para elevar as capacidades dos LLMs além das técnicas convencionais de instrução.

Impacto na Engenharia de Instruções

A engenharia de instruções, a arte de elaborar instruções para obter respostas desejadas dos LLMs, tem sido um aspecto crítico, porém frágil, do desenvolvimento de IA. A abordagem do DSPy para a instrução programática reduz significativamente a dependência da elaboração manual de instruções, mitigando assim os problemas associados à engenharia de instruções.

Redução da Fragilidade

As aplicações baseadas em LLM são propensas a degradação de desempenho quando ocorrem mudanças no LLM ou no pipeline de dados. O paradigma de programação sobre instrução do DSPy minimiza essa fragilidade, fornecendo um framework estável e sistemático para definir o comportamento do LLM. A capacidade de ajustar e otimizar as instruções programaticamente leva a aplicações mais resilientes (Monigatti).

Experiência do Desenvolvedor Aprimorada

A transição da engenharia de instruções para instrução programática oferece aos desenvolvedores uma abstração mais intuitiva e poderosa para a construção de aplicações. Com o DSPy, os desenvolvedores podem aproveitar suas habilidades de programação existentes para projetar interações LLM complexas, sem a necessidade de elaboração intricada de instruções. Essa mudança não só simplifica o processo de desenvolvimento, mas também abre novas possibilidades de inovação no campo das

aplicações GenAI (Datanami).

Conclusão

O DSPy representa uma mudança de paradigma no desenvolvimento de aplicações baseadas em LLM. Ao substituir a engenharia de instruções tradicional por um modelo centrado na programação, o DSPy aprimora a robustez, o desempenho e a experiência do desenvolvedor de sistemas de IA. Conforme o framework continua a evoluir, ele está pronto para redefinir o cenário da engenharia de instruções e pavimentar o caminho para aplicações LLM mais sofisticadas e confiáveis.

2.5 Function Calling e Output Parsers

Para a integração efetiva de aplicações potencializadas por Modelos de Linguagem de Grande Escala (LLMs), como a GPT, é crucial implementar interfaces de chamada de função (Function Calling) e mecanismos de análise de saída (Output Parsers). Estes componentes são fundamentais para a interoperabilidade entre sistemas de software tradicionais e as capacidades avançadas de processamento de linguagem natural oferecidas pelos LLMs. A interoperabilidade requer uma padronização na saída dos dados do processamento da LLM, para o qual o formato JSON é frequentemente preferido devido à sua ampla adoção e facilidade de uso em diferentes plataformas e linguagens de programação.

A chamada de função é um procedimento que permite a solicitação de serviços computacionais específicos do LLM, passando parâmetros de entrada de forma estruturada e recebendo um resultado que pode ser posteriormente processado ou apresentado ao usuário. Este paradigma é essencial para a modularidade e a reutilização de código, permitindo que o LLM seja invocado de maneira controlada e previsível.

Por outro lado, os analisadores de saída são projetados para interpretar e converter os dados brutos fornecidos pelo LLM em formatos utilizáveis por outros sistemas. Isso inclui a extração de informações relevantes, a transformação de estruturas de dados complexas em representações mais simplificadas ou a tradução de saídas em ações ou comandos específicos de software.

A adoção desses mecanismos é uma prática recomendada que promove a abstração, a manutenção e a escalabilidade em ambientes de desenvolvimento de software, oferecendo uma ponte robusta entre a inteligência artificial avançada e as aplicações do dia a dia. Garantir a compatibilidade e a comunicação eficiente entre esses sistemas distintos é um desafio que, quando superado, desbloqueia um novo horizonte de possibilidades em computação e tecnologia da informação.

2.6 Agentes

3 DESENVOLVIMENTO DE UMA LLM POWERED APPLICATION PARA FACILITAR O CONSUMO E APRENDIZAGEM ATRAVÉS DE VÍDEOS

3.1 Motivação

A aprendizagem ativa tem sido objeto de vários artigos acadêmicos importantes, destacando seu impacto positivo nos resultados dos alunos. A pesquisa de Freeman et al. (2014) demonstrou que a aprendizagem ativa pode aumentar significativamente as notas dos alunos em relação aos métodos didáticos, com alunos em cursos sem aprendizagem ativa sendo 1,5 vezes mais propensos a reprovar do que aqueles com aprendizagem ativa. Além disso, a aprendizagem ativa tem sido mostrada positiva para melhorar a motivação dos alunos, habilidades de pensamento crítico, retenção de informações e habilidades interpessoais. Por outro lado, a aprendizagem passiva, como assistir a vídeos, tem sido associada a desvantagens, como níveis mais baixos de engajamento e compreensão superficial de conceitos-chave.

Além disso, dado o contexto de excesso de dados e nossa crescente dificuldade em navegar em meio a um mar de informações, cada vez mais se faz necessário tecnologias que possam ajudar com esses problemas contemporâneos.

É importante ressaltar também que muitas vezes o conteúdo de interesse está disponível em outra língua, dificultando a aprendizagem. Apesar do Youtube possibilitar a geração de legendas automáticas, a legenda gerada por esse método não possui tanta qualidade quanto por exemplo com os modelos mais avançados de Reconhecimento de Voz Automático (ASR).

A criação de uma aplicação potencializada por LLMs e ASR, como o Whisper, tem o poder de transformar vídeos educacionais em experiências de aprendizagem mais ricas e interativas. Ao incorporar funcionalidades como a geração automática de capítulos, facilitação de perguntas e respostas por capítulos, e o uso de Retrieval Augmented Generation (RAG) para melhoria do QA em vídeos, esta aplicação não apenas melhora a acessibilidade e a personalização do conteúdo, mas também promove a aprendizagem ativa e engajada. Ademais, a transcrição e geração automática

de legendas tornam o conteúdo acessível para um público mais amplo, independentemente do idioma nativo do espectador, ampliando assim o alcance e a eficácia da educação por meio de vídeos.

3.2 Features

3.2.1 Improved Readability

3.2.2 Transcrição/Tradução com Whisper

3.2.3 Auto Chapter

3.2.4 Geração de perguntas por capítulo

4 CONCLUSÃO

REFERÊNCIAS

AGUIAR, M.; MARILTON, A. **Título da Monografia**. 2005. 85p. Trabalho de Conclusão (Curso de Ciência da Computação) — Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas.

MOORE, R. E. **Methods and Applications of Interval Analysis**. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1979. xi + 190p.

NEUMANN, J. von; AGUIAR, M. **Theory of Self-Reproducing Automata**. [S.l.: s.n.], 1966. xix + 388p.

Apêndices

APÊNDICE A – Um Apêndice

Anexos

ANEXO A – Um Anexo

Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla.

Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla.

ANEXO B – Outro Anexo

Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla.

Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla. Bla blabla blablabla bla.