

## AI 엔지니어링(AI Engineering)

인공지능(AI) 기술을 설계, 개발, 배포 및 유지 관리하는 모든 과정을 포괄하는 엔지니어링 분야입니다.

이는 AI 모델이나 시스템을 구현/활용할 수 있도록 기술적 기반을 마련하고, 안정적/효율적으로 작동하게 만드는 것을 목표로 합니다.

# 책 설명

- AI의 최근 혁신은 AI 제품에 대한 수요를 증가시켰을 뿐만 아니라 AI 제품을 구축하고자 하는 사람들의 진입 장벽을 낮추었습니다. 서비스로서의 모델 접근 방식은 AI를 난해한 분야에서 누구나 사용할 수 있는 강력한 개발 도구로 전환했습니다. 사전 AI 경험이 거의 없거나 전혀 없는 사람을 포함하여 모든 사람이 이제 AI 모델을 활용하여 애플리케이션을 구축할 수 있습니다. 이 책에서 저자인 Chip Huyen은 AI 엔지니어링, 즉 쉽게 사용할 수 있는 기반 모델을 사용하여 애플리케이션을 구축하는 프로세스에 대해 설명합니다.
- 이 책은 AI 엔지니어링에 대한 개요로 시작하여 전통적인 ML 엔지니어링과 어떻게 다른지 설명하고 새로운 AI 스택을 논의합니다. AI를 많이 사용할수록 치명적인 실패의 기회가 많아지고 따라서 평가가 더욱 중요해집니다. 이 책은 빠르게 성장하는 AI-as-a-judge 접근 방식을 포함하여 개방형 모델을 평가하는 다양한 접근 방식을 논의합니다.
- AI 애플리케이션 개발자는 모델, 데이터 세트, 평가 벤치마크, 그리고 겉보기에 무한한 수의 사용 사례와 애플리케이션 패턴을 포함하여 AI 환경을 탐색하는 방법을 알아봅니다. 간단한 기술로 시작하여 보다 정교한 방법으로 진행하는 AI 애플리케이션을 개발하기 위한 프레임워크를 배우고 이러한 애플리케이션을 효율적으로 배포하는 방법을 알아봅니다.
- AI 엔지니어링이 무엇이고 기존 머신 러닝 엔지니어링과 어떻게 다른지 알아보세요.
- AI 애플리케이션 개발 프로세스, 각 단계의 과제 및 이를 해결하기 위한 접근 방식을 알아보세요.
- 프롬프트 엔지니어링, RAG, 미세 조정, 에이전트, 데이터 세트 엔지니어링을 포함한 다양한 모델 적응 기술을 탐색하고 이러한 기술이 작동하는 방식과 이유를 이해합니다.
- 기초 모델을 제공할 때 지연 및 비용에 대한 병목 현상을 조사하고 이를 극복하는 방법을 알아보세요.
- 귀하의 요구 사항에 맞는 올바른 모델, 데이터 세트, 평가 벤치마크 및 메트릭을 선택하세요

# 교재 목차

## 1. Foundation Models를 이용한 AI 애플리케이션 구축 소개

AI 엔지니어링의 개념, 왜 주목받나, AI엔지니어링을 위해서 어떤 것들이 중요한가?

## 2. 기초 모델 이해

모델 훈련, 미세 조정, 샘플링

## 3. 평가 방법론

평가 방법론, 엔트로피 개념, 임베딩

## 4. AI 시스템 평가

평가 기준, 비용 및 시간, 모델 선택 워크플로, 평가 파이프라인 설계

## 5. 신속한 엔지니어링

시스템 프롬프트와 사용자 프롬프트, 컨텍스트 효율성, 신속한 엔지니어링 도구 평가

## 6. RAG 및 에이전트

RAG 건축, 검색 알고리즘 및 최적화

## 7. 미세 조정

미세 조정 기술, 미세 조정의 이유와 하지 않는 이유, 메모리 병목 현상, 양자화

## 8. 데이터셋 엔지니어링

데이터 수집 및 주석, 데이터 품질, 데이터 증강 / 합성 / 처리

## 9. 추론 최적화 (학습 이후 추론하는 프로세스를 개선, 속도/자원/비용 등)

추론 최적화, AI 가속기, 추론 서비스 최적화

## 10. AI 엔지니어링 아키텍처 및 사용자 피드백

AI 엔지니어링 아키텍처, AI 파이프라인 오케스트레이션, 사용자 피드백

# 1장. Foundation Models를 이용한 AI 애플리케이션 구축 소개

## 1. [AI 엔지니어링의 부상](#)

- [언어 모델에서 대규모 언어 모델로](#)
- [대규모 언어 모델에서 기본 모델로](#)
- [기본 모델부터 AI 엔지니어링까지](#)

## • [기본 모델 사용 사례](#)

- [코딩](#)
- [이미지 및 비디오 제작](#)
- [글쓰기](#)
- [교육](#)
- [대화형 봇](#)
- [정보 수집](#)
- [데이터 구성](#)
- [워크플로 자동화](#)

## • [AI 애플리케이션 계획](#)

- [사용 사례 평가](#)
- [기대치 설정](#)
- [마일스톤 계획](#)
- [유지](#)

## • [AI 엔지니어링 스택](#)

- [AI 스택의 3가지 계층](#)
- [AI 엔지니어링 대 ML 엔지니어링](#)
- [AI 엔지니어링 대 풀스택 엔지니어링](#)

## • [요약](#)

# 1장.

## 기본 모델을 활용한 AI 애플리케이션 구축 개론

# AI 엔지니어링의 부상

## • 기본 모델의 시작

- 언어 모델의 역사는 70년 이상
- 국소적 언어모델(확률적, 규칙 기반 접근법) → 초대규모 파라미터, 범용성, 멀티모달  
대규모 언어모델(심층신경망, Transformer 등) → 기본 모델 순으로 발전함
  - 대규모 모델을 학습시키고 운영하는 데 많은 자원(데이터, 연산 능력)이 필요
  - 특정 도메인에 특화되지 않으면 성능이 제한적
- 대표 애플리케이션: ChatGPT, GPT-4, LLaMA, PaLM, GitHub Copilot 등

# AI 엔지니어링의 현재와 미래

## • AI 애플리케이션

- 과거: 추천 시스템, 사기 탐지, 고객 예측 등
- 현재: 대규모 모델로 새로운 가능성 제공 (범용성, 확장성)

## • 변화의 요점

- 새로운 AI 스택: 손쉽게 활용 가능한 기본 모델 (사전 학습모델 활용)
- 전통적 ML 엔지니어와 달라진 AI 엔지니어의 역할
- 애플리케이션 패턴과 주요 고려사항 → 실행 전 필수 검토

## • 결론

- 기본 모델 부상 → AI 엔지니어링 폭발적 성장
- 도전과 기회 속에서 AI 애플리케이션 혁신 가능

AI 시스템을 구축할 때는 단순히 모델을 선택하는 것뿐 아니라 전체적인 애플리케이션 구조와 환경을 고려

# 언어 모델에서 대규모 언어 모델로

## • 언어 모델이란?

- 텍스트에서 통계적 정보를 인코딩하여 단어 또는 문맥 예측
- 활용 예: "My favorite color is \_\_" → 'blue'를 'car'보다 더 자주 예측

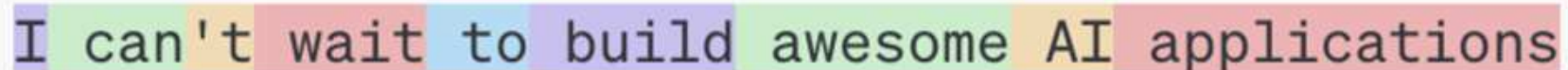
## • 역사적 배경

- 1905년: 소설 설록 홈즈, 영어 글자 통계를 활용
- 1951년: 클로드 새넌, "영어의 예측과 엔트로피" 발표

정보 이론의 관점에서 영어 언어의 예측 가능성과 엔트로피(무작위성)를 분석한 연구

## • 토큰나이제이션

- 문자, 단어 또는 단어의 일부로 세분화한 것을 토큰이라 함
- 텍스트를 토큰으로 나누는 과정을 토큰나이제이션(tokenization)



I can't wait to build awesome AI applications



# 언어 모델에서 대규모 언어 모델로

- 언어 모델의 두가지 주요 유형

- 마스크 언어 모델

- 문장에서 누락된 단어를 예측하도록 훈련

- 예를 들어, "My favorite \_ is blue(내가 가장 좋아하는 \_은 파란색입니다)"에서 빈칸 앞뒤의 문맥을 사용해 빈칸에 들어갈 단어를 예측

- 주로 감정 분석이나 텍스트 분류와 같은 작업에 사용됨

- 자가 회귀 언어 모델

- 이전 단어만 사용하여 시퀀스에서 다음 단어를 예측하도록 훈련

- 예를 들어, "My favorite color is \_ (내가 가장 좋아하는 색은 \_입니다)"에서 다음에 올 단어를 예측

- 오늘날 자가 회귀 언어 모델은 텍스트 생성 작업에서 선호

# 언어 모델에서 대규모 언어 모델로

- 자기 지도 학습(Self-supervision)

- 언어 모델이 스케일링 접근법(scaling approach)의 중심이 되었고, ChatGPT의 순간적 인기를 이끈 특별한 점은 무엇일까요?
  - 언어 모델은 자기 지도 학습(self-supervision)을 사용하여 훈련될 수 있는 반면, 많은 다른 모델들은 지도 학습(supervision)을 필요로 한다는 것
  - 자기 지도 학습은 데이터 레이블링 병목현상을 극복하여 모델이 학습할 수 있는 더 큰 데이터셋을 생성하는 데 기여했으며, 결과적으로 모델의 확장이 가능하도록 했습니다.
- 자기 지도학습의 장점
  - 지도학습은 데이터 레이블링에 비용이 많이 들고, 시간이 많이 소요됨
  - 수동으로 생성된 레이블을 사용해 모델을 훈련시키는 대신, 자연적으로 발생하는 레이블을 사용하여 모델을 훈련할 수 있음
  - 텍스트의 모든 시퀀스가 레이블 데이터로 사용될 수 있기 때문

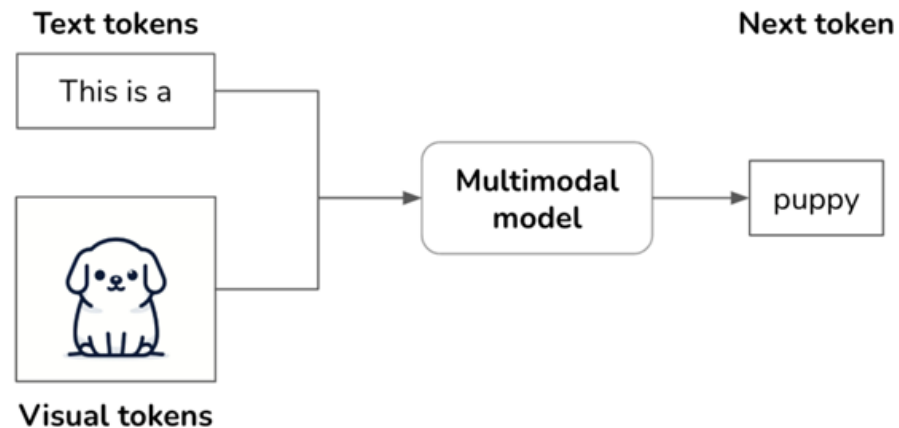
# 언어 모델에서 대규모 언어 모델로

Table 1-1. Training samples obtained from the sentence “I love street food.” for language modeling.

| Input (context)                 | Output (next token) |
|---------------------------------|---------------------|
| <BOS>                           | I                   |
| <BOS>, I                        | love                |
| <BOS>, I, love                  | street              |
| <BOS>, I, love, street          | food                |
| <BOS>, I, love, street, food    | .                   |
| <BOS>, I, love, street, food, . | <EOS>               |

# 대규모 언어 모델에서 기본 모델로

- 텍스트를 넘어선 데이터를 처리할 수 있는 능력은 AI가 현실 세계에서 작동하는 데 필수적임
  - 더 많은 데이터 모달리티를 통합하도록 확장 (텍스트+이미지+비디오+전문지식)
  - Gemini와 GPT-4V를 LLM으로 부르지만, 기본 모델로 발전함
  - 오랜 시간 동안 AI 연구는 데이터 모달리티 별로 나뉘어 있었지만 멀티모달 모델로 진화하고 있음



# 대규모 언어 모델에서 기본 모델로

- 자기 지도 학습(self-supervision)은 멀티모달 모델에서도 작동함
  - OpenAI는 자연어 지도 학습(natural language supervision) 이라는 자기 지도 학습 변형을 사용하여 언어-이미지 모델 CLIP (OpenAI, 2021)을 훈련
  - 각 이미지를 수동으로 레이블링하는 대신, 인터넷에서 함께 등장하는 (이미지, 텍스트) 쌍을 찾아서 훈련함 → CLIP CLIP은 생성형은 아니고 임베딩 모델임
  - CLIP은 Flamingo, LLaVA, Gemini와 같은 생성형 멀티모달 모델의 핵심

# 대규모 언어 모델에서 기본 모델로

- 기본 모델은 작업 특정 모델(task-specific model)에서 범용 모델(general-purpose model)로의 전환을 나타냄
  - 기본 모델은 그 크기와 훈련 방식 덕분에 광범위한 작업을 수행
  - 기본 상태에서도 범용 모델은 많은 작업에서 비교적 잘 작동할 수 있지만, 특정 작업의 성능을 극대화하기 위해 자주 "조정(tweaked)" 이 필요함
- 기본 모델의 성능 개선
  - 원하는 제품 설명의 유형에 대해 모델에 상세한 지침을 제공  
: 프롬프트 엔지니어링(prompt engineering)
  - 지침에 훌륭한 예시를 포함시킴  
: Few-shot 학습(few-shot learning)
  - 데이터베이스를 사용해 모델의 지침을 보완하는 방식  
: 검색 증강 생성(retrieval augmented generation, RAG)

# 대규모 언어 모델에서 기본 모델로

- **기본 모델이 없다면,**
  - 특정 작업에 대해 처음부터 모델을 훈련
  - 모델의 크기가 작다면 기존 대규모 모델만큼 성능이 뛰어나지 않을 수 있음
  - 훨씬 더 많은 시간과 데이터가 필요
- **일반적으로 기본 모델 은 AI 애플리케이션 개발 비용을 절감하고, 시장 출시 시간을 단축함**

# 기본 모델에서 AI 엔지니어링으로

- AI 엔지니어링이란 무엇인가?

- 기본 모델을 기반으로 애플리케이션을 구축하는 과정을 의미
- ML 엔지니어링(ML engineering) 또는 MLOps(ML operations)이 있는데 왜 지금 AI 엔지니어링에 대해 이야기할까?  
전통적인 ML 엔지니어링이 ML 모델 개발에서 시작된다면,  
AI 엔지니어링은 기존 ML 모델에서 시작함

- AI 엔지니어링의 성장을 위한 이상적인 3가지 요소에 대해서 알아보자



# 요소 1: AI 기능의 발전

- 강력한 기본 모델의 혁신적 기능

- 개방형 응답 생성 능력으로 다양한 작업 수행 가능성 확대
- 글쓰기, 디자인, 코드 작성 등 다양한 분야에서의 활용
- 사용자 기반 및 AI 애플리케이션 수요 증가 설명

## 요소2: AI 투자 증가

- AI 어플리케이션 개발 비용 저렴, 콘텐츠 출시 속도 빨라짐
  - AI에 대한 투자 수익률 향상
  - S&P 500 기업의 실적 발표에서 AI 언급 증가 (2023년 데이터 : 117 → 177개)

## 요소3: 어플리케이션 구축의 낮은 진입 장벽

- 대중화된 서비스형 모델 접근 방식

- 프로그래밍 언어 대신 프롬프트 엔지니어링으로 구현
- 강력하고, 범용적인 기본 모델
  - Google, Meta, Microsoft, Baidu, Tencent 와 같은 대기업
  - 일본, UAE 와 같은 정부
  - OpenAI, Mistral, Adept 와 같은 야심차고 자금이 잘 지원된 스타트업
- 2023년, 전 세계 웹 개발자 수는 약 1,900만 명으로 추정

# 기본 모델을 활용한 AI 애플리케이션

- 기본 모델을 활용한 AI 애플리케이션 사용 사례 (Use Cases)

- AWS는 기업의 생성형 AI 사용 사례를 세 가지 범주로 분류  
: 고객 경험, 직원 생산성, 프로세스 최적화
- 2024년 **O'Reilly** 설문 조사에서는 8가지로 분류  
: 프로그래밍, 데이터 분석, 고객 지원, 마케팅 카피, 기타 카피, 연구, 웹 디자인, 예술

# 기본 모델을 활용한 AI 애플리케이션

| 카테고리         | 소비자 사용 사례 예시 | 기업 사용 사례 예시        |
|--------------|--------------|--------------------|
| 이미지 및 비디오 제작 | 사진 및 비디오 편집  | 프레젠테이션             |
|              | 디자인          | 광고 생성              |
| 코딩           | 코딩           | 코딩                 |
| 글쓰기          | 이메일          | 이메일                |
|              | 에세이 편집       | 카피라이팅, SEO         |
| 교육           | 과외           | 온보딩 교육             |
|              | 에세이 채점       | 직원 역량 강화 교육        |
| 대화형 봇        | 일반 챗봇        | 고객 지원              |
|              | AI 동반자       | 제품 보조 기능           |
| 정보 집계        | 요약           | 요약                 |
|              | 문서 대화 기능     | 시장 조사              |
| 데이터 조직화      | 이미지 검색       | 지식 관리              |
|              |              | 문서 처리              |
| 워크플로우 자동화    | 여행 계획        | 데이터 추출, 입력 및 주석 작성 |
|              | 레스토랑 예약      | 리드 생성              |
|              | 이벤트 계획       |                    |

# 기본 모델을 활용한 AI 애플리케이션

- 205개의 오픈 소스 애플리케이션에 대한 이러한 사용 사례 분포

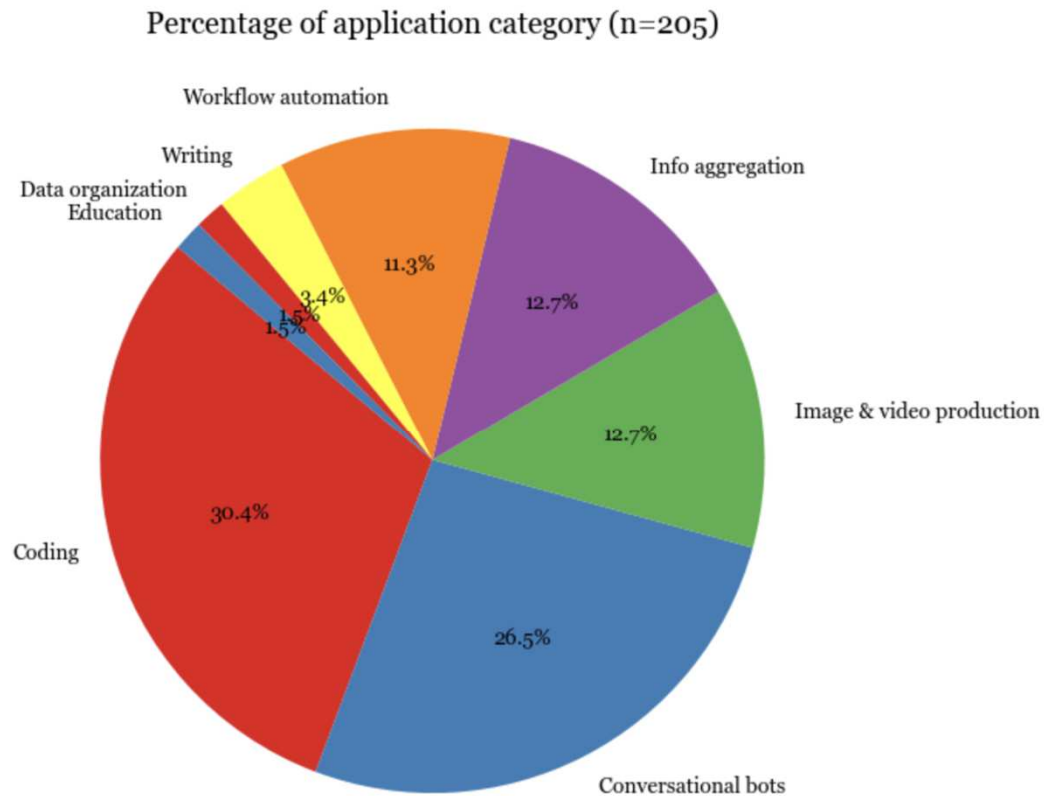


Figure 1-5. Distribution of use cases in the 205 open source repositories on GitHub.

코딩, 대화형 봇, 이미지/비디오 생성, 정보 집계, 상위권 교육, 데이터 조직화, 글쓰기 사용 사례의 비율이 낮음

- 기업은 외부 고객 지원 챗봇과 같은 외부 지향 애플리케이션보다 내부 지식 관리와 같은 내부 지향 애플리케이션을 더 빠르게 배포함

- 기본 모델은 개방형이며 어떤 작업에도 사용할 수 있지만, 이를 기반으로 구축된 많은 애플리케이션은 여전히 종료형(close-ended)

# 기본 모델을 활용한 AI 애플리케이션

- 코딩

- AI가 코딩에 능숙함
- AI 애플리케이션의 초기 개발자들이 주로 코딩 문제에 많이 노출됨

- **GitHub Copilot, Devin, gpt-engineer, screenshot-to-code 등이 인기있음**

- **AI가 소프트웨어 엔지니어들의 역할을 완전히 대체할 것인가?**

- 문서화 작업, 코딩작업은 AI가 많이 유용하며, 복잡한 역할일수록 개선폭이 적음

# 기본 모델을 활용한 AI 애플리케이션

- 특정 코딩 작업에 특화된 많은 도구

- 웹페이지와 PDF에서 구조화된 데이터를 추출하는 코드를 생성 (AgentGPT)
- 영어를 코드로 변환 (DB-GPT, sqlchat, PandasAI)
- 디자인, 스크린샷을 기반으로 이미지를 렌더링하는 웹사이트 코드를 생성 (screenshot-to-code, draw-a-ui)
- 하나의 프로그래밍 언어나 프레임워크를 다른 것으로 변환 (gpt-migrate, ai-code-translator)
- 문서 작성 (autodoc)
- 테스트 생성 (PentestGPT)
- 커밋 메시지 생성 (aicommits)



# 기본 모델을 활용한 AI 애플리케이션

- 이미지 및 비디오 제작

- AI는 창의적인 작업에 적합함
- 개인 이미지 개선 뿐만 아니라 기업에서 광고/마케팅 분야에 빠르게 적용 중

- 글쓰기

- 더 나은 의사소통을 위해서 AI를 활용함 (메일/책 쓰기 등)
- Google Docs, Notion, Gmail과 같은 노트 작성 및 이메일 앱
- 글쓰기 보조 앱인 Grammarly (일관성을 위해서 미세 조정함)

# 기본 모델을 활용한 AI 애플리케이션

- **교육**

- 교육에서 AI를 거부하기 보다는 더 잘 활용할 수 있도록 하는 문화 형성 중
- 수업 교재 및 보조자료로 활용

- **대화형 봇 (Conversational bots)**

- 상담사, 동료, 비서 등의 역할이 가능함

# 기본 모델을 활용한 AI 애플리케이션

- 정보 통합 (Information aggregation)

- 메시지, 뉴스, 자료, 일정 등의 정보를 통합/요약함
- AI는 웹사이트, 연구를 요약하고 사용자가 선택한 주제에 대한 보고서를 작성
- 생성형 AI 사용자 중 74%가 아이디어 간소화, 정보 요약한다는 조사 결과

- 그 외

- 데이터 조직화, 워크플로 자동화

# AI 애플리케이션을 구축할 때 고려사항

- **빠른 변화의 속도**

- 모델의 한계 개선, 성능 향상, 비용 저감  
(비용 문제로 자체개발했는데, 얼마 지나지 않아서 상용모델이 더 저렴해짐)
- 대신 규제, 법률적이 미확립되어서 혼란이 발생할 수 있음 (데이터 소유권 등)

- **제품 방어력(Product Defensibility)**

- 낮은 진입 장벽은 축복이자 저주 (특히 대기업 서비스 vs 벤처기술)

# AI 엔지니어링 스택(The AI Engineering Stack)

- AI 엔지니어링은 ML(Machine Learning) 엔지니어링에서 발전하였으며, 역할은 AI 엔지니어링과 ML 엔지니어링을 구분해야함
- AI 스택의 세 가지 계층

## Application development

평가, 프롬프트 엔지니어링, AI 애플리케이션 인터페이스 제공을 포함

Prompt engineering  
AI interface  
RAG  
Evaluation

## Model development

모델 개발을 위한 도구를 제공

모델링, 훈련, 파인튜닝, 추론 최적화, 데이터셋 엔지니어링을 위한 프레임워크

Dataset engineering  
Inference optimization  
Modeling & training

## Infrastructure

모델 제공, 모니터링, 데이터와 컴퓨팅 관리 등을 위한 도구를 포함

Compute management  
Data management  
Serving  
Monitoring

# AI 엔지니어링 스택(The AI Engineering Stack)

GitHub에서 최소 500개의 별을 받은 업체의 스택 분석

Cumulative count of repos by category over time

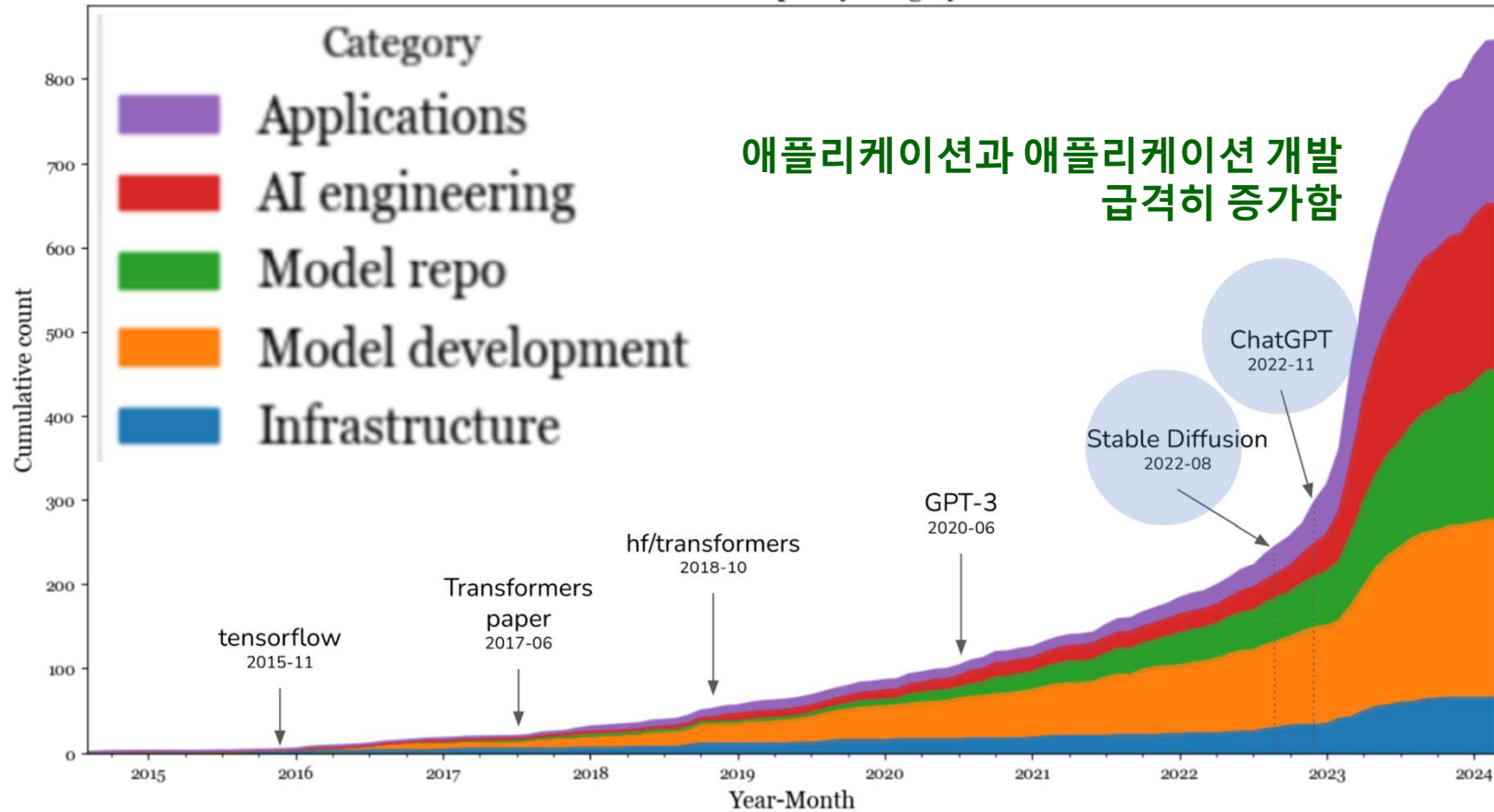


Figure 1-12. Cumulative count of repositories by category over time

# AI 엔지니어링 vs. ML 엔지니어링

- 기존의 ML 엔지니어링과 다른 주요 3 가지

- 기본 모델이 없을 경우,

AI 엔지니어링은 모델 개발보다는 모델을 적응시키고 평가하는 데 더 중점을 둠

AI 엔지니어링에서는 다른 사람이 이미 훈련한 모델을 사용함

즉, 모델링과 훈련보다 프롬프트 엔지니어링과 파인튜닝 등 모델 적응 기술에 집중

- AI 엔지니어링은 기존 ML 엔지니어링보다 더 큰 모델을 다룸

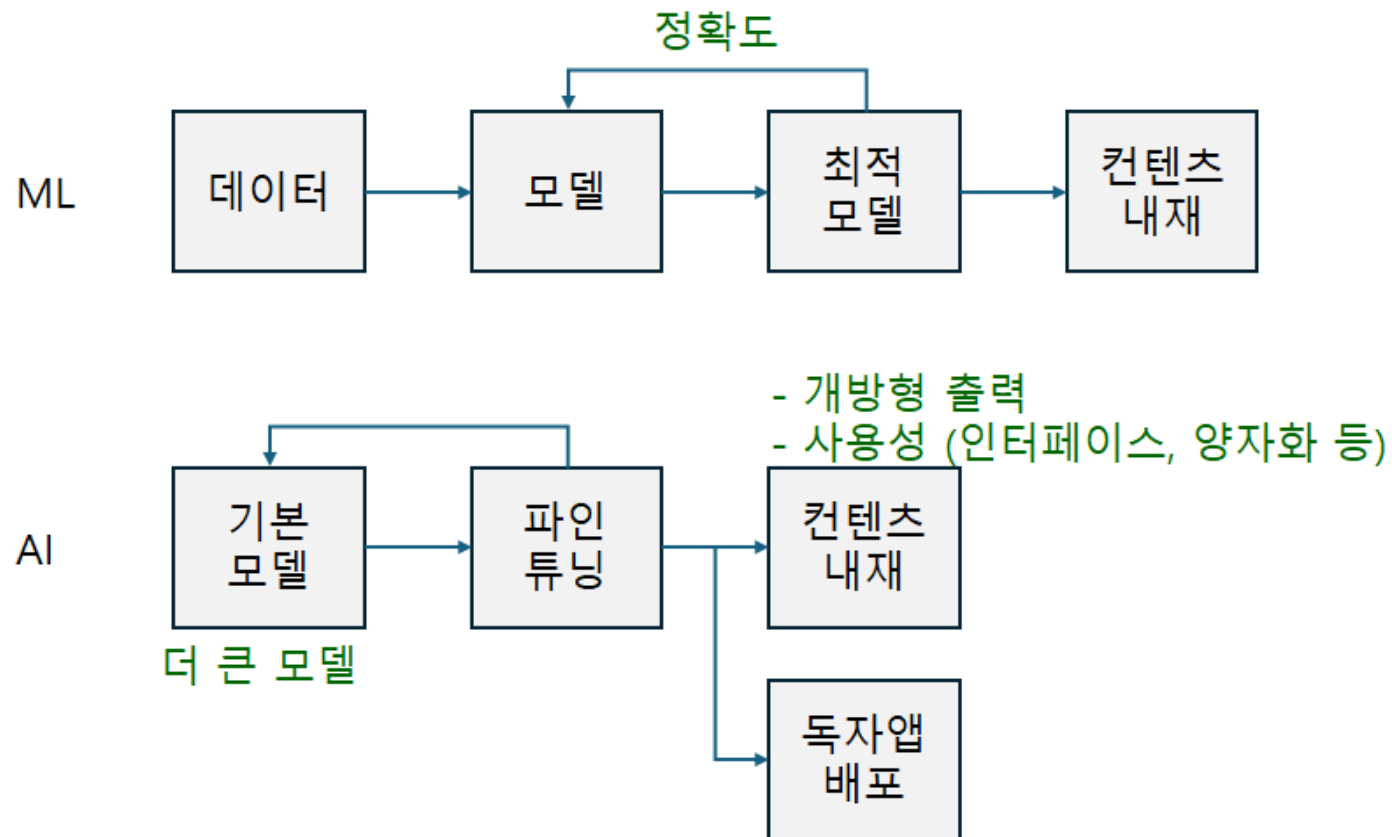
더 큰 모델은 다루기 더 어렵고, 더 많은 컴퓨팅 리소스와 연산 시간을 사용  
더 효율적인 훈련과 추론 최적화 중요도 높음

GPU와 대형 클러스터를 다룰 줄 아는 엔지니어의 필요성이 증가했음을 의미함

- 개방형 출력을 생성할 수 있는 모델을 다룸



# AI 엔지니어링 vs. ML 엔지니어링



# 모델 적용 기술

## • 프롬프트 엔지니어링

- 모델 자체를 변경하지 않고, 컨텍스트 입력을 제공하여 모델을 적응시킴  
RAG(정보 증강 검색)도 프롬프트 엔지니어링의 한 부분임
- 시작하기 쉽고, 적은 데이터를 필요로 함
- 하지만, 엄격한 성능이 필요한 경우에는 충분치 않을 수 있음

## • 파인튜닝과 추론 최적화

- 모델 가중치를 업데이트 하는 기술
- 모델의 품질과 속도 측면에서 크게 향상시킬 수 있음

# 모델 개발(Model development)

- 모델 개발은 전통적인 ML 엔지니어링과 가장 많이 연관된 계층
- 모델링 및 훈련, 데이터셋 엔지니어링, 추론 최적화로 구성됨
  - 모델 아키텍처를 설계/훈련/파인튜닝하는 과정임  
(도구 예 : Google의 TensorFlow, HuggingFace의 Transformers, Meta의 PyTorch 등)
  - 전문적인 ML 지식이 필요함
- 사전 훈련(Pre-training): 처음부터 모델을 훈련하는 것으로, 모델의 가중치는 무작위로 초기화 됨
- 파인튜닝(Finetuning): 이전에 훈련된 모델을 계속해서 훈련하는 과정임

# AI 엔지니어링 vs. ML 엔지니어링

- **데이터셋 엔지니어링(Dataset engineering)**

- AI 모델 훈련에 필요한 데이터를 수집, 정제, 변환, 저장, 라벨링하는 작업과 특징 엔지니어링
- 필요한 데이터의 양 : 초기 모델 > 파인튜닝 > 프롬프트 엔지니어링
- ML 엔지니어링 : 특징 엔지니어링 중심
- 기본 모델 : 전처리, 중복 제거, 토큰화, 데이터 품질 향상에 더 중점을 줌

- **추론 최적화(Inference optimization)**

- 모델을 더 작고, 빠르고, 더 저렴하게 만드는 것을 의미함
- 기법 : 양자화(quantization), 증류(distillation), 분해(factorization), 그리고 더 빠른 디코딩을 위한 모델 변경 등
- 기본 모델이 등장하면서 추론 최적화의 중요성은 더욱 커짐

# AI 엔지니어링 vs. ML 엔지니어링

- 기본 모델과 함께 모델 개발의 범주가 어떻게 변화했는가

| 범주         | 전통적인 ML에서의 구축                      | 기본 모델에서의 구축                                |
|------------|------------------------------------|--|
| 모델링 및 훈련   | 처음부터 모델을 훈련하려면<br>ML 지식이 필요함       | ML 지식은 있으면 좋지만<br>필수는 아님                   |
| 데이터셋 엔지니어링 | 특징 엔지니어링에 중점을 둠<br>특히 테이블형 데이터의 경우 | 특징 엔지니어링보다<br>데이터 전처리, 토큰화, 품질 관리에 더 중점을 둠 |
| 추론 최적화     | 중요함                                | 더 중요함                                      |

### **비구조화 데이터 (Unstructured Data):**

정해진 형식이 없는 데이터로, 텍스트, 이미지, 비디오, 오디오 파일 등이 포함됩니다. 예를 들어, 소셜 미디어 게시물, 이메일, 문서 파일 등이 이에 해당합니다.

### **반구조화 데이터 (Semi-structured Data):**

일정한 구조는 있지만, 고정된 스키마가 없는 데이터입니다. XML, JSON, HTML 파일 등이 대표적입니다. 이러한 데이터는 태그나 키-값 쌍을 사용하여 정보를 표현합니다.

### **시계열 데이터 (Time Series Data):**

시간에 따라 변화하는 데이터를 나타내며, 주식 가격, 기온 변화, 센서 데이터 등이 포함됩니다. 이 데이터는 시간 순서에 따라 정렬되어 분석됩니다.

### **그래프 데이터 (Graph Data):**

노드(정점)와 엣지(간선)로 구성된 데이터로, 소셜 네트워크, 추천 시스템 등에서 사용됩니다. 이 형식은 관계를 시각적으로 표현하는 데 유용합니다.

### **멀티미디어 데이터 (Multimedia Data):**

이미지, 비디오, 오디오 등 다양한 형식의 데이터를 포함합니다. 이러한 데이터는 주로 콘텐츠 관리 시스템이나 미디어 플랫폼에서 사용됩니다.

# AI 엔지니어링 vs. ML 엔지니어링

- **애플리케이션 개발(Application development)**

- 전통적인 ML 엔지니어링 : 팀이 독자적인 모델을 사용해 애플리케이션을 구축하며, 모델의 품질이 차별화 요소가 됨
- 반면, 기본 모델을 사용하는 경우 : 여러 팀이 동일한 모델을 사용하므로, 차별화는 애플리케이션 개발 과정을 통해 이루어져야 함
- 애플리케이션 개발 계층 구성
  - 평가(evaluation)
  - 프롬프트 엔지니어링(prompt engineering)
  - AI 인터페이스(AI interface)

# AI 엔지니어링 vs. ML 엔지니어링

- 평가(Evaluation)

- 평가의 두가지 목표

- 기본 모델을 평가하여 작업에 가장 적합한 모델을 선택하는 것 (적합성)
    - 선택한 모델을 적응시키는 과정에서의 진행 상황을 평가하는 것 (작동성)

- 평가는 ML 엔지니어링에서 항상 중요했지만, 기본 모델에서는 그 중요성이 더 커짐

- 기본 모델의 출력이 개방형이기 때문 (ML은 비교할 수 있는 예상 결과가 명확하지만, 챗봇,/이미지생성 등의 출력의 정확도를 평가하는 것은 쉽지 않음)
  - 모델의 용량과 AI 엔지니어링 기술이 계속 발전하고 있기 때문 (벤치마크와 실제 상황이 다를 수 있음)



# AI 엔지니어링 vs. ML 엔지니어링

- **프롬프트 엔지니어링**

- 프롬프트 엔지니어링 : 모델의 가중치를 변경하지 않고, 컨텍스트만을 사용해 AI 모델이 원하는 행동을 수행하게 만드는 기술
- 단순히 모델에 올바른 지시를 제공하는 것뿐만 아니라, 구조화된 출력을 생성하는 작업, 데이터베이스를 사용해 모델의 컨텍스트를 확장하는 작업(RAG), 메모리 관리 (과거 질의응답 기억) 등을 포함
- Gemini 평가 사례에서 프롬프트 엔지니어링 기술로 Gemini Ultra 의 MMLU 성능을 83.7% 에서 90.04% 로 향상

# AI 엔지니어링 vs. ML 엔지니어링

- AI 오케스트레이션

- AI 오케스트레이션 도구는 애플리케이션의 실행 단계를 지정할 수 있도록 해줌
- 많은 팀은 기존 워크플로우 오케스트레이션 도구(예: Airflow, Dagster, Metaflow)를 활용해 생성형 AI 작업을 지원함
- 많은 팀은 LangChain, Dify, Flowise와 같은 생성형 AI 작업에 특화된 오케스트레이션 도구로 시작함

# AI 엔지니어링 vs. ML 엔지니어링

- AI 인터페이스(AI interface)

- 최종 사용자가 AI 애플리케이션과 상호작용할 수 있도록 인터페이스를 만드는
- 기본 모델이 등장하면서 비교적 쉽게 AI 애플리케이션을 구축 가능하게 됨
- AI 애플리케이션은 독립형 제품일 수도 있고 다른 제품에 통합(ex. 추천시스템)될 수도 있음
- AI 애플리케이션에서 점점 인기를 얻고 있는 인터페이스의 몇 가지 예시
  - 웹 및 데스크톱 애플리케이션
  - 브라우저 확장 프로그램
  - 봇 (Slack, Discord, WeChat, WhatsApp 등과 같은 채팅 애플리케이션에서 작동)
  - 플러그인 (개발자가 VSCode, Shopify, Microsoft Office와 같은 애플리케이션에 AI를 통합 지원)

# AI 엔지니어링 vs. ML 엔지니어링

- AI 인터페이스(AI interface)

| 범주         | 전통적 ML 구축 | 기본 모델을 활용한 구축 |
|------------|-----------|---------------|
| AI 인터페이스   | 덜 중요      | 중요            |
| 프롬프트 엔지니어링 | 적용되지 않음   | 중요            |
| 오케스트레이션    | 중요        | 중요            |
| 평가         | 중요        | 더 중요          |

# AI 엔지니어링 vs. 풀스택 엔지니어링

- 인터페이스에 대한 강조가 증가하면서 프론트엔드 엔지니어링에 대한 수요가 높아짐
- 프론트엔드 전문성의 중요성이 커지면서 AI 도구의 API에도 변화가 생김
  - 전통적으로 ML 엔지니어링은 파이썬(Python) 중심
  - 오늘날 JavaScript API에 대한 지원도 증가하고 있음  
LangChain.js, Transformers.js, OpenAI의 Node 라이브러리, Vercel의 AI SDK 등
- 이러한 변화는 AI 엔지니어링을 풀스택 개발에 더 가깝게 함
  - “AI 엔지니어링은 AI 모델이 추가된 소프트웨어 엔지니어링일 뿐이다.”
- 풀스택 엔지니어
  - 아이디어를 빠르게 데모로 구현하고, 피드백을 받고, 가장 유망한 아이디어를 반복 개선할 수 있음
  - 먼저 제품을 구축하고, 제품이 유망하다고 판단되면 그때 데이터와 모델에 투자하는 것이 가능해짐

# AI 엔지니어링 vs. 풀스택 엔지니어링

- 풀스택 엔지니어링(Full Stack Engineering)은 웹 애플리케이션의 프론트엔드와 백엔드 모두를 다룰 수 있는 능력을 가진 소프트웨어 엔지니어를 의미함
  - 즉, 사용자가 직접 보거나 상호작용하는 프론트엔드와 서버에서 데이터 처리 및 비즈니스 로직을 담당하는 백엔드 모두를 개발할 수 있는 역량을 갖춘 개발자
- 풀스택 엔지니어의 주요 역할
- 주요 업무
  - 프론트엔드 개발: HTML, CSS, JavaScript 등으로 사용자 인터페이스(UI)를 구축하고 디자인  
React나 Vue 같은 프레임워크를 사용해 동적인 웹 페이지를 개발
  - 백엔드 개발: 서버 측에서 데이터베이스와 상호작용하고 비즈니스 로직을 구현함  
Node.js나 Python으로 서버를 구축하고, RESTful API를 설계하여 프론트엔드와 데이터를 주고받음
  - 통합 및 배포: 다양한 플랫폼과 서비스 간의 통합 작업 수행하여, Docker와 같은 도구로 컨테이너화하고 배포 파이프라인 설정
- 풀스택 엔지니어는 프론트엔드와 백엔드 간의 소통 문제를 줄이고 프로젝트 전반에서 독립적으로 작업할 수 있음

# AI 엔지니어링 vs. 풀스택 엔지니어링

| 구분    | AI 엔지니어링   | 풀스택 엔지니어링  |
|-------|--|--|
| 정의    | AI 엔지니어링은 인공지능 시스템을 설계, 개발 및 배포하는 과정입니다. 이는 머신러닝, 딥러닝, 자연어 처리(NLP) 등 다양한 AI 기술을 포함                                   | 풀스택 엔지니어링은 웹 애플리케이션의 프론트엔드와 백엔드 모두를 개발할 수 있는 능력을 가진 엔지니어링 분야입니다.   |
| 주요 기술 | 머신러닝 알고리즘 (예: 회귀, 분류, 클러스터링)<br>딥러닝 프레임워크 (예: TensorFlow, PyTorch)<br>데이터 전처리 및 분석 (예: Pandas, NumPy)<br>모델 평가 및 최적화 | 프론트엔드 기술 (예: HTML, CSS, JavaScript, React, Vue.js)<br>백엔드 기술 (예: Node.js, Django, Ruby on Rails)<br>데이터베이스 관리 (예: MySQL, MongoDB)<br>API 설계 및 통합 |
| 역할    | AI 엔지니어는 데이터 과학자와 협력하여 데이터셋을 준비하고, 모델을 훈련시키며, AI 솔루션을 실제 환경에 배포합니다.  | 풀스택 엔지니어는 전체 애플리케이션의 구조를 이해하고, 사용자 인터페이스에서부터 서버와 데이터베이스까지 모든 부분을 개발하고 유지 관리합니다.  |

전문성: AI 엔지니어는 AI 및 데이터 과학에 중점을 두고, 풀스택 엔지니어는 웹 개발의 모든 측면을 다룸

기술 스택: AI 엔지니어는 주로 데이터와 알고리즘에 집중하는 반면, 풀스택 엔지니어는 다양한 프로그래밍 언어와 프레임워크를 사용하여 애플리케이션을 구축함

업무 범위: AI 엔지니어는 AI 모델 개발과 관련된 작업을 수행하고, 풀스택 엔지니어는 전체 애플리케이션의 설계 및 구현을 담당함

# AI 엔지니어링 vs. 풀스택 엔지니어링

Traditional ML  
Data/ML engineers

Data → Model → Product

LLM enabled AI  
AI engineers

Product  $\xrightarrow{\text{if successful}}$  Data  $\xrightarrow{\text{if scaling}}$  Model

먼저 제품을 구축하고, 제품이 유망하다고 판단되면 그때 데이터와 모델에 할 수도 있음