

EleutherAI에서의 1년.

고현웅

kevin.ko@tunib.ai

1. 발표자 소개

1. 발표자 소개



안녕하세요! 🙌

저는 자연어처리, 초거대모델, 오픈소스를 사랑하는 개발자 고현웅(Kevin Ko)입니다.

(현) TUNiB Co-Founder & ML Engineer

(현) EleutherAI Lead ML Scientist

(전) KakaoBrain ML Engineer

추신. Kevin Ko는 제 회사 영어 이름인데, 외국인들이 '현웅' 발음을 너무 어려워해서 (혜능? 효능?) EleutherAI에서도 이 이름을 쓰고 있습니다.

2. EleutherAI는 어떤 곳인가요?

2. EleutherAI는 어떤 곳인가요?

◎ EleutherAI는...

- 초거대 언어모델을 연구하는 비영리 연구단체.
- GPTNeo, GPTJ, GPTNeoX를 만든 그룹.
- 즉, EleutherAI는 회사가 아니라 사이드프로젝트 모임 같은 것.
- 마음 맞는 사람이라면 누구나 참여해서 프로젝트를 할 수 있음.

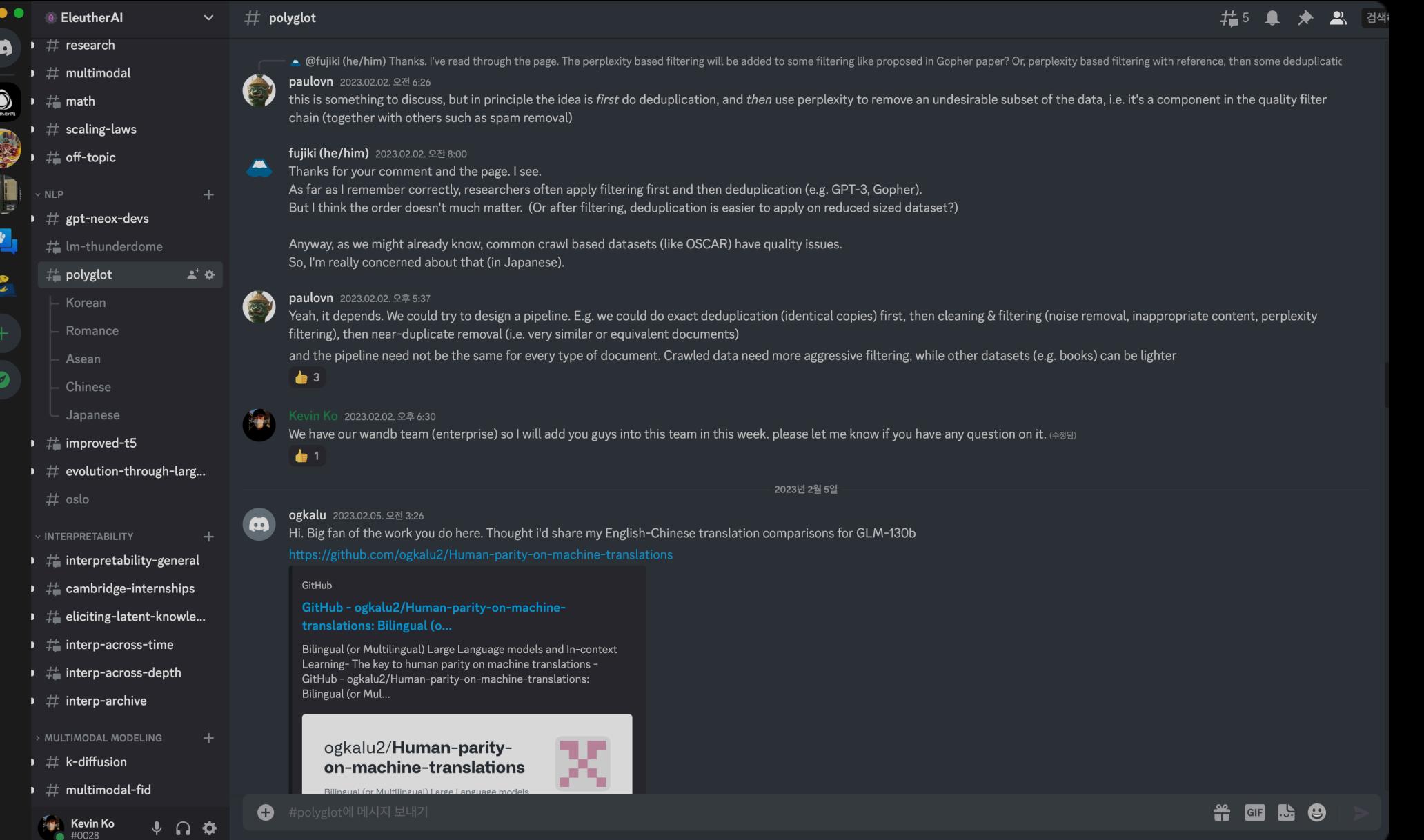


상상 속 모습



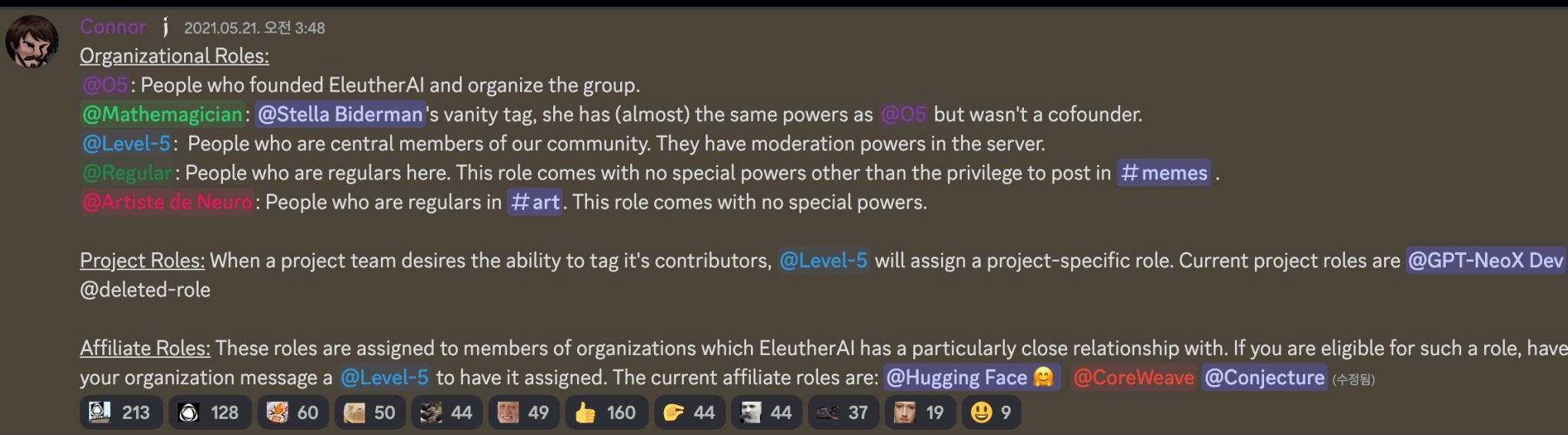
현실 속 모습

2. EleutherAI는 어떤 곳인가요?



- 주로 디스코드에서 운영되고 있음.
- 그 외 깃헙/노션 등을 활용하여 협업 진행.

2. EleutherAI는 어떤 곳인가요?



- 계급체계(?)가 있음. 발표자 본인은 Regular 랭크임.
- staff, research-lead 등 계급 외의 권한요소도 있음.

2. EleutherAI는 어떤 곳인가요?

다음과 같은 프로젝트들이 진행 중임.

- 1) Pythia: LLM 해석 [프로젝트](#).
 - 2) Polyglot: 멀티링구얼 언어모델 [프로젝트](#).
 - 3) Improved-T5: T5 부흥 [프로젝트](#).
 - 4) GPT-NeoX: Megatron 기반 LLM 학습툴킷 개발 [프로젝트](#)
 - 5) OSLO: HF 기반 LLM 학습툴킷 개발 [프로젝트](#)
 - 6) LM-Eval-Harness: LLM 평가툴킷 개발 [프로젝트](#)
- ... 등등

2. EleutherAI는 어떤 곳인가요?

그러면 GPU는 어떻게...?

stability.ai

킹갓-Stability AI에서 모든 장비를 무상으로 제공 중

3. Polyglot

<https://github.com/EleutherAI/polyglot>

3. Polyglot



Kevin Ko

EleutherAI에 들어온지도 벌써 2개월 쯤 되었군.
근데 살펴보니까 여기 GPU 장비 겁나 많은데!?
한국어 빅모델 만들어서 오픈소스화 하고싶은데...

TMI: 발표자는 딥러닝 병렬처리 라이브러리 개발을
하던 중에 협업 제안이 와서 합류하게 되었습니다.

3. Polyglot



Kevin Ko

근데 한국어 모델만 땀랑 만든다고 하면

임팩트가 너무 없어보일 것 같고...

멀티링궐 프로젝트를 해본다고 주장해보자..!

(한국어 모델도 같이 스리슬적...만들면 낫배드!?)

3. Polyglot



Kevin Ko

결국 research-lead 권한을 얻을 수 있었고,
곧바로 멤버를 모집하여 프로젝트에 돌입하였음.
우여곡절이 있었지만, polyglot-ko 모델을 개발할 수 있었음.

3. Polyglot

3.1. 데이터 수집

튜닝에서 열심히 수집한 1.2TB의 한국어 데이터 사용.

데이터 관련 자세한 내용은 보안상 공유 불가.

Source	Size (GB)	Link
Korean blog posts	682.3	-
Korean news dataset	87.0	-
Modu corpus	26.4	corpus.korean.go.kr
Korean patent dataset	19.0	-
Korean Q & A dataset	18.1	-
KcBert dataset	12.7	github.com/Beomi/KcBERT
Korean fiction dataset	6.1	-
Korean online comments	4.2	-
Korean wikipedia	1.4	ko.wikipedia.org
Clova call	< 1.0	github.com/clovaai/ClovaCall
Naver sentiment movie corpus	< 1.0	github.com/e9t/nsmc
Korean hate speech dataset	< 1.0	-
Open subtitles	< 1.0	opus.nlpl.eu/OpenSubtitles.php
AIHub various tasks datasets	< 1.0	aihub.or.kr
Standard Korean language dictionary	< 1.0	stdict.korean.go.kr/main/main.do

전처리 이후 860+GB 수준이 됨.

3. Polyglot

3.2. 데이터 전처리

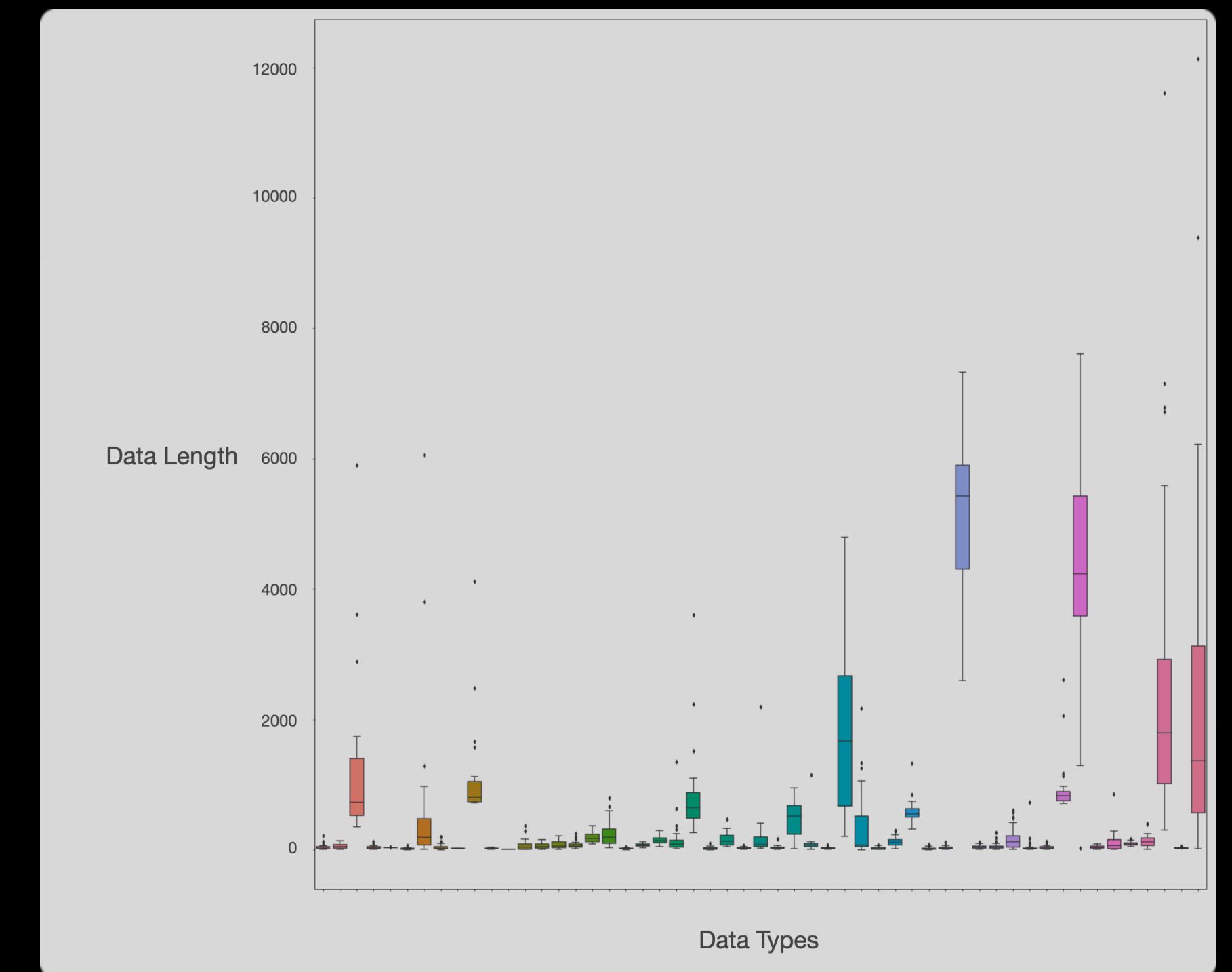
DeepMind의 Gopher에서 사용했던
Massive Text Filtering에서 소개된 기법들과 몇가지 한국어 전처리 로직 적용.
전체 코드베이스는 pyspark로 구현: <https://github.com/EleutherAI/dps>

1) doc_len_filter:

도큐먼트의 길이가 너무 길거나 짧은 경우 필터링.
적절한 길이를 선택하기 위해 데이터셋별 길이 분포를 조사하였음.

2) mean_word_len_filter:

도큐먼트에 포함된 단어들의 길이 평균이
특정 범위 내에 들어오지 못하는 경우 필터링.



데이터셋별 길이 분포, 대부분이 짧은 데이터에 속했음.

3. Polyglot

3.2. 데이터 전처리

DeepMind의 Gopher에서 사용했던

Massive Text Filtering에서 소개된 기법들과 몇가지 한국어 전처리 로직 적용.

전체 코드베이스는 pyspark로 구현: <https://github.com/EleutherAI/dps>

3) symbol_to_word_ratio_filter:

기호(!?@#\$%^&* 등)의 개수와 일반 단어의 개수 사이의

비율이 너무 높은 경우 필터링 (기호가 너무 많은 경우에 해당.)

4) bullet_ellipsis_filter:

도큐먼트 내에 불릿 포인트 (*, ., •, ·, ·)로 시작하는 라인(\n기준)이나

말 줄임표 (...)로 끝나는 라인의 비율이 높으면 필터링.

3. Polyglot

3.2. 데이터 전처리

DeepMind의 Gopher에서 사용했던

Massive Text Filtering에서 소개된 기법들과 몇가지 한국어 전처리 로직 적용.

전체 코드베이스는 pyspark로 구현: <https://github.com/EleutherAI/dps>

5) alphabetic_word_ratio_filter:

알파벳이 컬한 문자의 수가 너무 적으면 필터링.

6) reduce_emoticon:

ㅋㅋㅋㅋ, ㅎㅎㅎㅎㅎ 등 이모티콘이 과도한 경우 이를 줄임.

3. Polyglot

3.2. 데이터 전처리

DeepMind의 Gopher에서 사용했던

Massive Text Filtering에서 소개된 기법들과 몇가지 한국어 전처리 로직 적용.

전체 코드베이스는 pyspark로 구현: <https://github.com/EleutherAI/dps>

7) PII converting:

전화번호, 주민번호 등 개인정보는 <|tel|>, <|rrn|> 등의 토큰으로 마스킹.

8) Deduplication:

빠른 진행을 위해 Exact Match Deduplication 사용

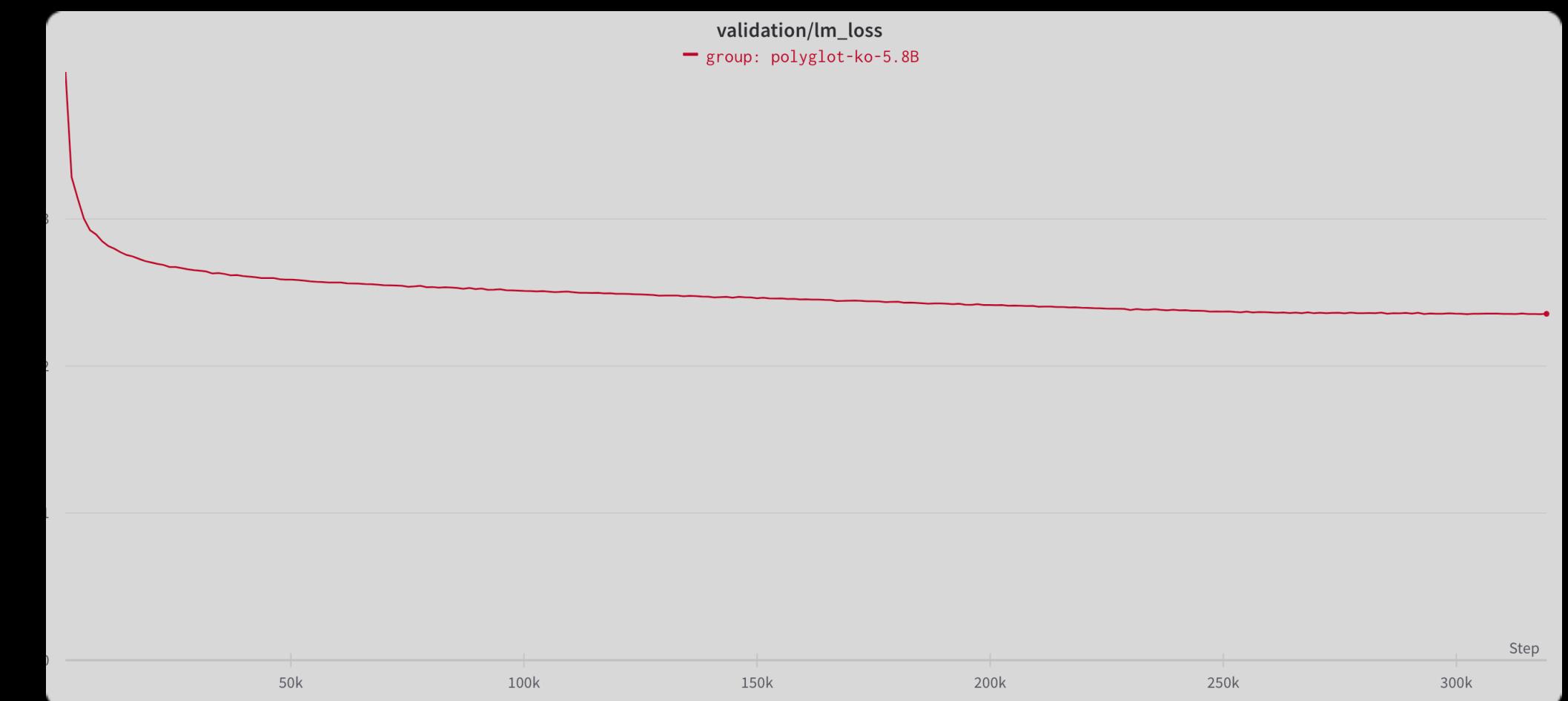
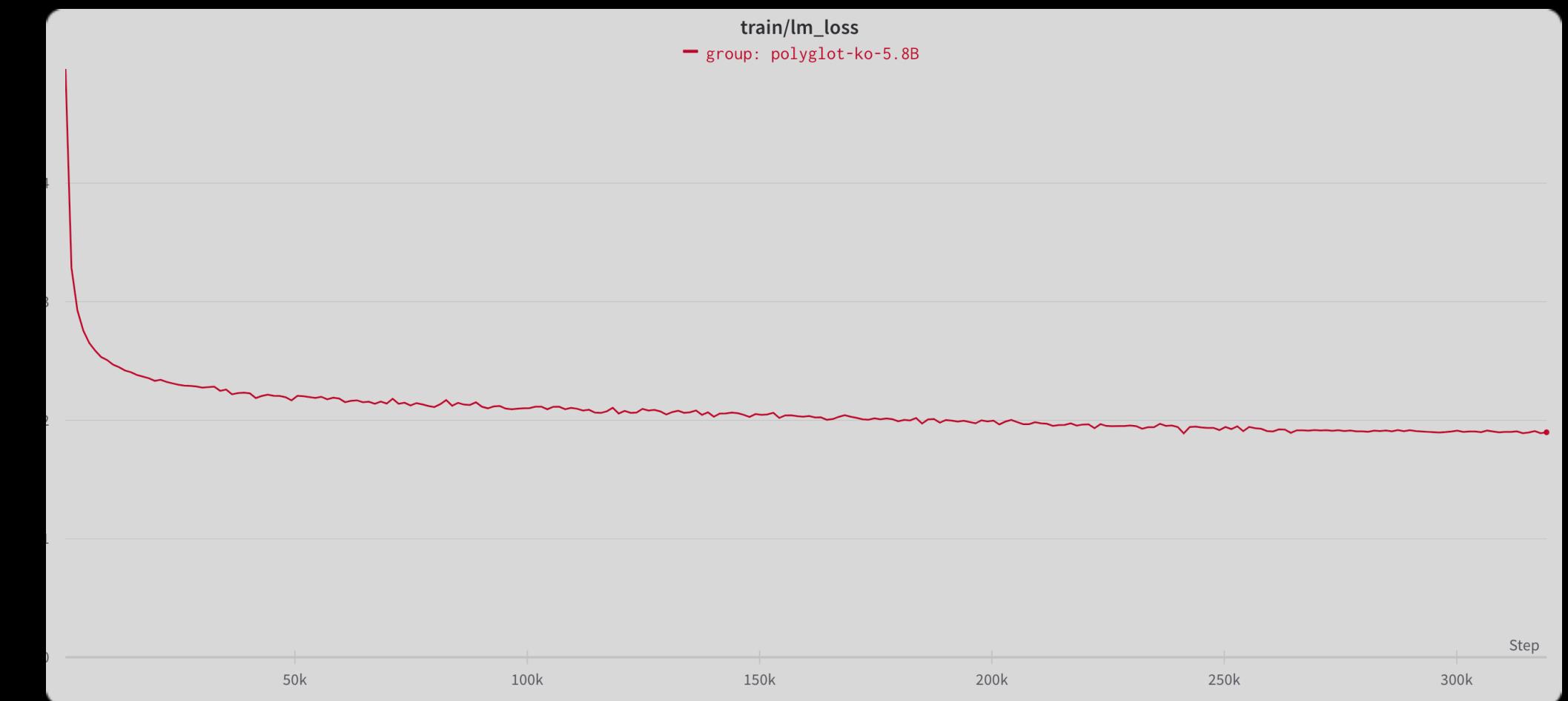
3. Polyglot

3.3. 모델 학습

EleutherAI의 GPT-NeoX 코드베이스를 활용하여 학습.

총 256개의 A100 GPU를 이용하여 3개 모델을 각각 학습시켰음.

- 1.3B: 213B의 토큰을 102,000 스텝 학습, 약 1~2주 소요.
- 3.8B: 219B의 토큰을 105,000 스텝 학습, 약 2~3주 소요.
- 5.8B: 172B의 토큰을 320,000 스텝 학습, 약 3~4주 소요.



3. Polyglot

3.4. 모델 평가

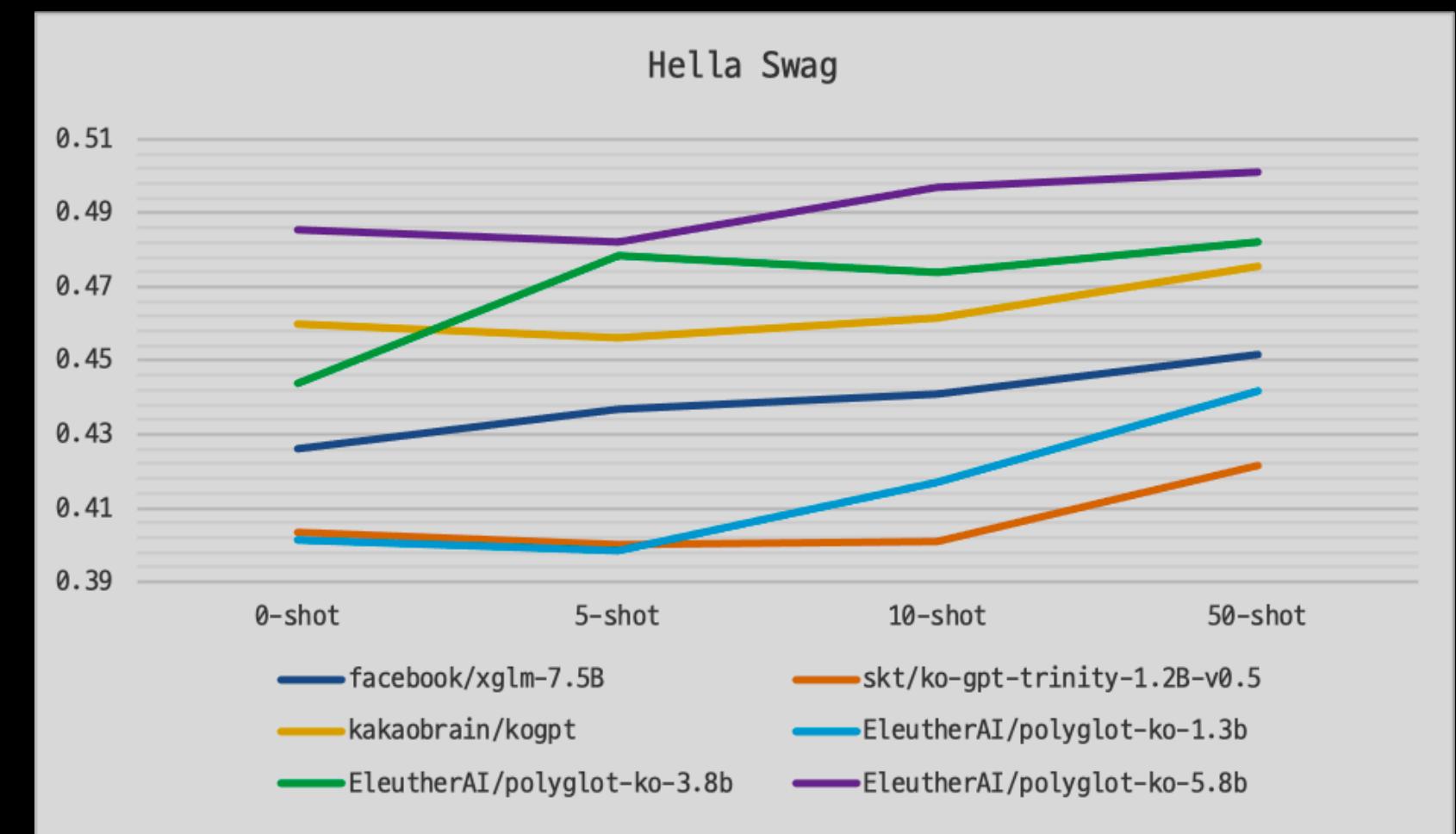
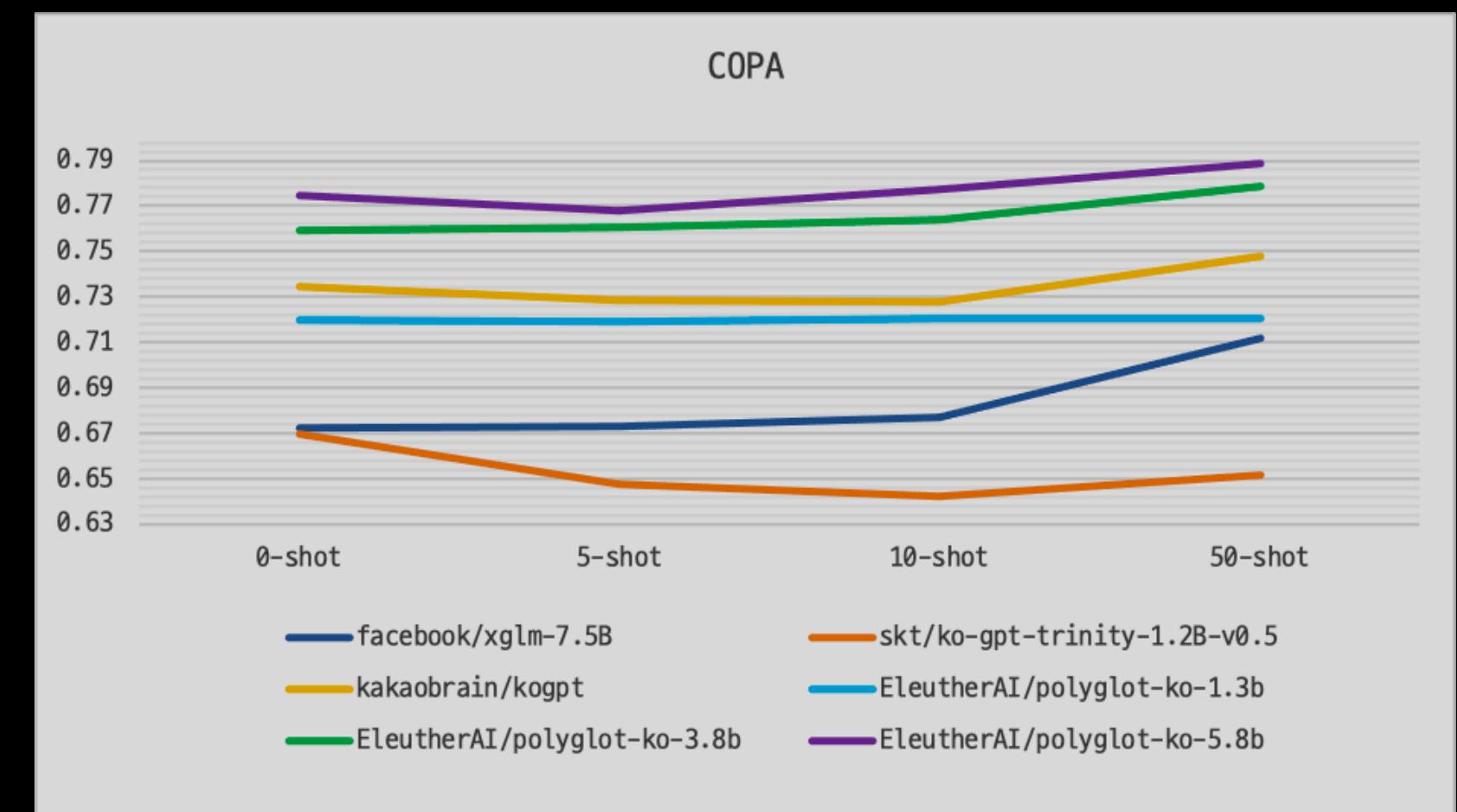
EleutherAI의 LM-Eval-Harness에 한국어 태스크 추가.

모델 평가 데이터셋으로는 SKT의 KOBEST 활용.

논문에 제시된 프롬프트와 동일한 것을 사용하여 few-shot 생성능력 평가.

모델 카드에는 KOBEST의 5개의 태스크 중 COPA와 HellaSwag만 공개했는데, 다른 태스크에서 작성했던 Evaluation 코드에 버그가 있는 것으로 발견되어 결국에 사용하지 않음. 당시 Evaluation 코드를 작성하신 분이 액티브하게 활동하기 어려우신 상황이여서 수정이 어려웠음.

개발한 모델들이 두 태스크에서 대체적으로 가장 좋은 성능을 기록하였고 현재 Hugging Face에 "EleutherAI/polyglot-ko-5.8b"와 같은 이름으로 배포 되어있음.



3. Polyglot

3.5. Polyglot-ko-v2

Polyglot-ko 12.8B 학습 중에 프로젝트를 조금 재정비 하기로 결정하였고

이후부터 지금까지는 Polyglot-ko-v2를 준비하고 있음.

1.3B부터 다시 학습을 시작하여 최종적으로 40B까지 확장하는 것을 목표로 함.

변경사항:

1. 원천 데이터의 양을 기준 대비 약 2배로 증가 (2+TB)

Chinchilla 논문에 의하면 10B 모델은 기존보다 훨씬 많은 데이터를 필요로 함.

2. 전처리 로직 강화

- Exact Match Deduplication을 Min Hash 기반으로 변경하여 중복제거 성능 개선

- 개인정보 처리 방식 개선: 단순히 토큰으로 대체하면 모델이 개인정보의 패턴을 이해 할 수 없음.

따라서, 기존에 <|tel|>로 대체되던 부분이 <|tel_start|>랜덤전화번호<|tel_end|>와 같은 방식으로 변경됨.

- 많은 한국어 전처리 로직 추가, 특히 뉴스 데이터 전처리 능력이 크게 개선되었음.

3. Polyglot

3.5. Polyglot-ko-v2

Polyglot-ko 12.8B 학습 중에 프로젝트를 조금 재정비 하기로 결정하였고

이후부터 지금까지는 Polyglot-ko-v2를 준비하고 있음.

1.3B부터 다시 학습을 시작하여 최종적으로 40B까지 확장하는 것을 목표로 함.

변경사항:

3. 고품질 데이터 추가

기존에 학습하지 못했던 문학, 수능교재, 국회의사록, 깃헙 데이터 (주석번역) 등

다른 모델들은 학습하기 어려운 유니크하고 퀄리티 높은 데이터를 대거 수집하여 학습데이터에 추가

4. Evaluation 개선

기존에 문제가 있던 Evaluation 코드를 전부 고치고 KOBEST 이외의 평가 데이터셋도 다양하게 추가하여
한국어 LLM 평가 시스템의 기준과 같은 위치에 자리매김 하기.

3. Polyglot

3.6. Polyglot-asian / Polyglot-romance

Polyglot은 본래 멀티링구얼 프로젝트이므로 멀티링구얼 연구도 진행중.

멀티링구얼 모델을 잘 만들기 위한 근본적인 솔루션을 찾아나가는 연구가 될 것임.

예를 들면 모델 사이즈와 수용 가능한 언어의 수 사이의 관계를 찾는다는 등..

이미 Stability 산하에 인도지역 언어모델을 만드는 프로젝트가 있기 때문에 인도를 제외하고, 인구수가 높은 편에 속하는 총 11개의 언어 (한/중/일/베트남/타이/인도네시아/말레이/영어/스페인/프랑스/이탈리아/포르투갈/루마니아) 데이터를 수집했으며, 현재까지 각 언어당 최소 200~300GB, 최대 2TB까지 수집하였으며 총 10TB 이상의 데이터를 수집하였음.

멀티링구얼 모델링 계획:

- 2월까지 데이터 수집 마무리 하기
- 3월부터 전처리 들어가기
- 4월부터 연구/모델학습 시작

4. 오픈소스 그룹 매니징

4. 오픈소스 그룹 매니징

장점 (vs기업):

- 고급인력을 영입하기 훨씬 쉬운 편이다.

단점 (vs기업):

- 50명을 영입하면 그 중에서 10명 정도만 실제로 활동한다.

깃헙 뱃지 때문에, 이력서에 한 줄을 추가하려고, 혹은 신년 다이어트 다짐 같은 마음으로 참여 하는 멤버가 대다수.

- 급여와 같은 구속력이 없다. 동료가 언제든 팀을 떠날 수 있다.
- 사람들은 (1)여유롭고 (2)원할때만 일을 한다.
- 특정 태스크를 특정 멤버에게 할당했는데, 그 멤버가 안하거나 탈퇴하면 해당 태스크가 진행이 안된다.
- 특정 멤버에게 업무를 강요 할 수 없다.

4. 오픈소스 그룹 매니징



Stas 오전 3:20

To manage an open source project you need to understand the nature of open source - as I have been involved in Open Source project for now more than 25 years you need to understand that Open Source is all about people scratching their itch. It's basically people who want to solve a problem that they either can't find solved already elsewhere or they don't like how it is solved. So they go ahead and solve it (or try to).

If you then ask those folks to do more work beyond what they are excited about it often doesn't work, as they are no longer motivated to do so.

Some people want to contribute to a large successful Open source project due to fame of being associated and bragging rights.

Most Open Source contributors feed on the thank yous they receive from the users of their work. This is invaluable currency that feeds our egos and some of us can do an amazing amount of work just for a few thanks. That's our human nature.

Now that you understand the motivation, the best way to have others do the work for you in an open source project is

- to do everything to enable them do their work. e.g. if the person is a great coder but hates documentation or testing, don't ask them to do the latter - find someone else to help them with it. Otherwise they will just disappear
- find a way for their work to be visible and praise a lot (but in honest way, not a manipulative)
- thank them personally for their contributions and gently ask them if there are any other items that might be interested to work on, but affirm that nothing is binding - often having a nice TODO list broken down into categories of easy/med/hard and perhaps by topic makes it easy for people to choose their tasks.
- as far as dead lines go, this is a difficult one - usually when people are motivated by their own desire they find time. When I started with Open Source contributing I was doing it in my free time because I was excited about it. So having a full time job wasn't an issue.

There are probably many other things to share, but in general make it easy for people to do the work they are good at and make it exciting to have the accomplishments.

You can also learn a lot by contributing to other open source projects, you will quickly see what works and what doesn't

some projects for example (like `pytest`) I avoid like plague since a few of the key developers are extremely obnoxious and aggressive and I avoid sending PRs there or even Issues. (편집됨)

EAI is actually in that group too, some of their developers are extremely difficult to communicate with - i.e. super-unfriendly and aggressive - that's why there is little collaboration between HF and EAI

things have warmed up a lot through big science collaboration.

4. 오픈소스 그룹 매니징

오픈소스 매니징 10계명.

- 1) 기본적으로 해당 멤버가 왜 이 프로젝트를 하려는지 파악하고 있어야 한다.
- 2) 해당 멤버가 싫어하는 업무는 시키면 안된다. 그러지 않으면 그 멤버는 사라진다.
- 3) 그들의 작업물을 지속적으로 칭찬하고 홍보해줄 필요가 있다. (SNS든, 인터넷 커뮤니티든 어디든)
- 4) 그들이 기여하면 감사를 표하고 또 다른 업무를 해줄 수 있는지 부드럽고 조심스레 묻는다.
- 5) 보기 좋은 TODO 리스트를 만들어놓으면 기여자들이 본인들이 할 수 있는 일을 찾아서 하기 쉽다.
- 6) 기본적으로 내 열정의 크기와 다른 멤버들의 열정의 크기가 같다고 생각해선 안된다.
- 7) 리더 스스로가 열정을 잃은 모습을 보여서는 안된다. 리더가 열정에 찬 모습일 때 비로소 멤버들도 따라온다.
- 8) 일정은 디테일하게 잡지 않는다. 또한 그 일정을 너무 신뢰하지 말자. 어차피 잘 안지켜진다.
- 9) 일반적인 경우라면 하나의 업무에 2명 이상의 멤버를 투입하자. 한 멤버에게만 맡기면 그 멤버가 사라졌을 때 대처하기 어렵다.
- 10) 멤버들이 다 나가버려도 내가 포기하지만 않으면 다시 시작 할 수 있다.



감사합니다.