JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 26, No. 3, pp. 591-597, Mar. 2025

Check for updates

웹 페이지 디자인을 위한 대규모 언어 모델 비교 분석

이 태 준[1] · 박 정 석[2] · 한 개[3] · 정 회 경[4*]
[1]배재대학교 컴퓨터공학과 박사과정
[2]배재대학교 스마트ICT융합학과 박사과정
[3]우송대학교 융합경영학과 조교수
[4]배재대학교 컴퓨터공학과 교수

Comparative Analysis of Large Language Models for Web Page Design

Tae-Jun Lee[1] · Jung-Seok Park[2] · Kai Han[3] · Hoe-Kyung Jung[4*]
[1]Doctor's Course, Department of Computer Engineering, Paichai University, Daejeon 35345, Korea
[2]Doctor's Course, Department of Smart ICT Convergence, Paichai University, Daejeon 35345, Korea
[3]Assistant Professor, Department of Convergence Management, Woosong University, Daejeon 34606, Korea
[4]Professor, Department of Computer Engineering, Paichai University, Daejeon 35345, Korea

[요 약]

본 논문에서는 2024년에 공개된 대규모 언어 모델(LLM)과 WebSight 데이터셋을 활용하여 웹 디자인에 적용할 수 있는 모델의 성능을 비교·분석하고, 이를 기반으로 비즈니스 모델로의 활용 가능성을 탐구하였다. DeepSeek 모델은 코딩 특화 모델로 가장 낮은 손실 값을 기록하며 추론에서 우수한 성능을 보였으나, 웹 프로그래밍 언어 비율이 낮아 추가적인 데이터셋 보완이 필요하였다. Qwen 2.5 일반 모델은 코딩 특화 모델 대비 높은 일반화 성능과 유연성을 보였으나, 속도가 느리고 긴 출력 시 메모리 사용량이 증가하는 한계가 나타났다. LLaMA 3.2는 가장 적은 연산량과 빠른 추론 속도를 기록했지만, 제한된 데이터셋으로 인해 비교적 높은 손실 값과 부정확한 결과를 보였다.

[Abstract]

In this paper, we conducted a comparative analysis of the performance of models that can be applied to web design by utilizing the Large-Scale Language Model (LLM) and WebSight dataset released in 2024. Furthermore, we investigated the potential of utilizing them as a business model. The DeepSeek model is a coding-specific model that recorded the lowest loss value and showed excellent inference performance. However, the proportion of web programming languages was low, requiring additional dataset supplementation. The Qwen 2.5 general model showed high generalization performance and flexibility compared to coding-specific models; however, it had limitations, such as slow speed and increased memory usage when outputting long data. LLaMA 3.2 recorded the lowest computational requirements and fastest inference speed but exhibited relatively high loss values and inaccurate results because of the limited dataset.

색인어 : 대규모 언어 모델, 로컬 환경, 미세조정, 오픈 소스, 웹 디자인
Keyword : Large Language Model, Local Environment, Fine-Tuning, Open-Source, Web Design

http://dx.doi.org/10.9728/dcs.2025.26.3.591

Received 5 February 2025; **Revised** 25 February 2025
Accepted 28 February 2025

***Corresponding Author; Hoe-Kyung Jung**
Tel: +82-42-520-5640
E-mail: hkjung@pcu.ac.kr

Ⅰ. 서 론

최근 LLM(Large Language Model)의 발전으로 다양한 비즈니스 환경에서 이를 적용하고자 하는 연구가 활발히 이뤄지고 있다[1]. LLM이 수학 수식, 혹은 자연어, 프로그래밍 언어를 이용해 코딩하고 이를 디버깅(debugging)하는 도구로써 활용되고 있다[2].

현재 공개된 LLM의 경우 특정 목적에 맞도록 학습하기 위해 한계가 존재한다. 방대한 파라미터 개수로 인해 로컬 환경에서 모델을 학습하고 추론하는데 컴퓨팅 자원이 많이 들고 있어 외부에서 상용 서비스되고 있는 API를 활용해야 하는 경제적 어려움이 있다. 아울러 데이터셋을 수집하기 위해서는 데이터셋 수집 윤리나 법적 문제에 한계가 있다[3].

본 논문에서는 현재 2024년에 공개되어 있는 LLM과 공개된 데이터셋을 활용해 웹 디자인에 적용할 수 있는 모델을 로컬 환경에서 서로 비교하고 이를 고찰해 비즈니스 모델에 적용하기 위한 사전 연구로써 활용하고자 한다.

Qwen 2.5의 일반 모델과 코딩 모델, LLaMA 3.2, DeepSeek 모델을 비교한 결과 DeepSeek 모델이 가장 높은 결과를 보였고, Qwen 2.5 일반화 모델이 코딩 모델보다 더 높은 결과를 보였다. 그러나 로컬 환경에서 학습 시 파라미터가 더 큰 모델은 학습 시간이 매우 오래 걸리며, 비록 경량화 모델을 활용했더라도 추론 시 출력 코드 결과가 길어지면 컴퓨팅 자원 한계의 문제와 맥락과 관계없는 답변을 보이는 것으로 나타났다. 이는 기존 사전학습(pre-trained) 모델에서 이미 학습된 데이터의 불균형 문제와 평가 방식이 상이하다는 문제가 있는 것으로 보여진다.

본 논문의 구성은 다음과 같다. 2장에서는 본 연구에 활용한 LLM 모델과 미세조정 기법에 관해 관련 연구를 기술하고 3장에서는 실험에 활용한 데이터셋과 환경, 실험 설계 및 성능 평가 방법에 관해 기술한다. 이어 4장에서는 실험 결과와 기존 모델을 활용한 점에 관한 한계를 기술하고 마지막 5장에서는 결론을 기술한다.

Ⅱ. 관련 연구

2-1 LLM

현재 공개된 LLM 모델은 소스를 공개하지 않거나 공개하는 경우가 있으며, 공개되어 있는 경우라도 상세히 어떤 학습 기술이 활용되었는지, 학습 파라미터(parameter) 설정값을 공개하지 않고 있다. 다른 모델과 성능 비교만을 다루거나 어떤 데이터를 학습에 활용했는지만 공개되어 있으며 이는 학술지에 공개된 것이 아닌 모델 제작사에서 발행한 블로그나 기술문서(technical report), 깃허브(github) 코드로 공개되어 있다. 본 절에서는 2024년에 공개한 모델로 연구에 활용한 4가지 모델에 관해 간략히 기술한다.

먼저 LLaMA(Large Language Model Meta AI) 모델은 Meta AI 팀에서 개발한 LLM으로 현재 버전은 3까지 있으며 버전 1에서는 비교적 작은 크기의 모델을 통해 GPT-3보다 더 좋은 성능을 보였다[4]. GPT-3의 경우 1,750억 개를 가지고 있다면 LLaMA 1은 최대 650억 개다. 버전 2의 경우 모델 확장성과 다국어 지원에 초점을 뒀으며 버전 3의 경우 마이너 업그레이드를 통해 파라미터 개수 증가, 멀티모달 강화, 온디바이스(on-device) 기능이 추가됐다[5],[6].

Qwen(Qianwen) 모델은 Alibaba 그룹에서 공개한 모델로 파라미터 수는 18억, 70억, 140억 개의 모델을 지원해 LLaMA2 보다 더 좋은 성능을 보였으며 수학, 코딩에 특화된 모델을 따로 지원한다[7],[8].

DeepSeek 모델은 DeepSeek사와 베이징대학교가 합작해 만든 코딩에 특화된 모델로 깃허브에 오픈소스로 공개된 코드 데이터를 활용하였다[9]. 특히 코드를 프로젝트 레벨의 코딩 시나리오를 분류해 학습했으며 87개의 프로그래밍 언어와 2조 개의 토큰이 활용되어 13억, 330억 개의 파라미터 버전을 지원한다.

2-2 미세조정 기법

LLM은 파라미터 개수가 많으므로 모든 파라미터 학습(full-training)이 컴퓨팅 자원이 충분하지 않으면 불가능하다. 따라서 사전 학습 모델에서 일부 파라미터만을 수정하는 학습을 진행해야 한다. 이러한 기법을 PEFT(Personalized Fine-Tuning)라고 하며, 2가지 방식이 있다.

LoRA(Low-Rank Adaptation)는 Microsoft AI팀이 개발한 기법으로 사전 학습 모델을 동결(freeze)하고 낮은 랭크 행렬을 추가해 성능을 조정하는 방식이다[10],[11].

QLoRA(Quantized LoRA)는 LoRA를 기반으로 모델 파라미터를 4비트 혹은 8비트로 양자화하여 GPU와 TPU의 메모리 사용량을 줄이고 속도를 향상한 기술을 말한다[12]. 이에 관한 파라미터 설정은 3-3절에서 기술한다.

Ⅲ. 실험 데이터셋과 설계

3-1 실험 데이터셋

본 연구에 활용하기 위해 데이터셋으로는 웹 페이지 구성 요소인 HTML과 CSS 코드가 필요하고 사용자 동적 이벤트를 처리하기 위한 JavaScript 코드가 필요하다. 이는 여러 웹 페이지를 크롤링(crawling)하여 수집할 수 있지만 이 과정에서 발생하는 디자인권 문제, 크롤링 과정에서 과도한 트래픽을 유발하여 업무 방해의 문제가 있어 활용하지 못하였다.

이러한 이유로 인해 본 연구에서는 오픈소스(open

source)로 공개되어 있는 데이터셋을 활용하고자 한다. AI 모델과 데이터셋을 연구 목적으로 활용하고자 한 플랫폼인 HuggingFace에 공개된 WebSight 데이터셋이며 웹 페이지의 이미지, 코드, 웹 페이지의 내용과 형태 혹은 레이아웃에 관한 설명을 제공한다[13]. 이에 관한 주요 특징은 표 1과 같으며 예시는 그림 1과 같다.

표 1. WebSight 데이터셋의 특징
Table 1. Features of the WebSight dataset

	Description
Column composition	- Web page image - Web page code (HTML, CSS, JavaScript) - Description of the web page generated by LLM (content, form, layout, etc.)
Row count	- 1.92M
Version	- v1. Code including HTML/CSS, no images - v2. Code including HTML/Tailwind CSS, with images

그림 1. WebSight 데이터셋 예시
Fig. 1. WebSight dataset example

본 연구에서는 이미지, 동영상, 텍스트, 음성 등 다양한 형태의 데이터를 혼합한 학습 모델인 멀티모달(multi-modal) 모델이 아니므로 v2 버전의 이미지가 있는 데이터셋을 학습할 수 없으므로 v1 버전의 데이터셋을 활용하였다.

학습의 효율성을 높이고 컴퓨팅 자원을 절약하기 위해 데이터 샘플링(sampling)은 전체 1.91M 개의 데이터에서 무작위로 10,000개의 데이터를 추출해 실험에 활용하였다.

3-2 실험 설계

그림 2는 모델 학습에 전반적인 모식도이다.

먼저 WebSight v1 데이터셋을 활용해 4개의 모델을 미세조정하고, 최종 평가(evaluation)를 수행한다. 모델은 2장에서 언급한 모델 4개를 선정하였으며 모두 2024년에 공개된 모델이다. 아울러 모델 이름에서 "(Coder)"로 명시된 모델은 코딩에 특화된 모델이고, 명시되어 있지 않으면 일반화된 모델이다.

LLM 모델을 학습할 수 있는 플랫폼인 LLaMA-Factory를 사용했다. 모델 4비트 양자화 및 PEFT 기법(LoRA, QLoRA)이 적용된 최적화된 메모리 구조를 제공과 하드웨어

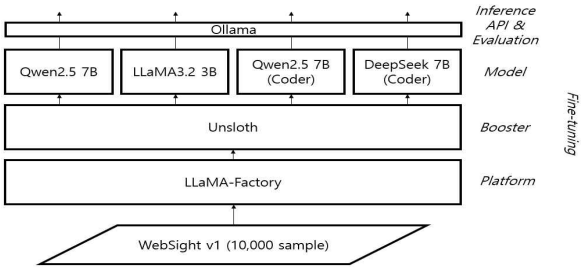

그림 2. 모델 학습 모식도
Fig. 2. Model learning schematic diagram

호환성을 맞추기 위한 Unsloth 라이브러리를 활용하였다. 학습된 모델을 로컬 환경에서 평가하고 추론하기 위해 Ollama도 함께 활용하였다. 표 2는 하드웨어 환경을 정리한 것이다.

표 2. 하드웨어 환경
Table 2. Hardware environment

Item	Specification
CPU	AMD Ryzen 7 5800X (x8 core, 4.5GHz)
RAM	32GB
GPU	Nvidia Geforce RTX 3080 (GDDR6X 10GB, 1.71GHz)
Storage	500GB SSD + 6TB HDD

3-3 모델 학습

앞서 언급한 4가지 모델에 관해 동일한 환경과 동일 파라미터를 설정해 진행하며 설정값은 표 3과 같다.

표 3. 학습 파라미터 설정
Table 3. Setting learning parameters

	Qwen 2.5	LLaMA 3.2*	Qwen 2.5 (Coder)	DeepSeek (Coder)
Number of parameters (model)	7B	3B*	7B	7B
Fine-tuning type	QLoRA	LoRA*	QLoRA	QLoRA
Quantized bits	4	-*	4	4
Quantized type	bitsandbytes			
Operation type	bf16 (float 16)			
Sequence length	1024			
LoRA rank	8			
Lora alpha (parameter output)	16			
LoRA dropout	0			
Learning rate	5e-5			
Learning rate scheduler	cosine			
Optimizer	AdamW (8bit)			
Epochs	3			

표 3에서 기울임으로 '*' 표시된 LLaMA 3.2는 파라미터 수가 경량화 모델로 1B, 3B밖에 없으며, 그 이상인 11B, 90B인 모델은 멀티모달 모델이다. 따라서, LLaMA 3.2는 파라미터 수가 3B인 모델로만 쓸 수 있으며 QLoRA를 하지 않고, LoRA로만 하여도 충분히 학습된다. 이에 따라 양자화 비트는 당연히 없다.

양자화 타입으로 bitsandbytes 라이브러리를 사용하였다. 이는 HuggingFace에서 개발한 경량화 라이브러리로 GPU에서 저비트(low bit) 양자화 연산을 지원하며 메모리 사용량을 대폭 줄일 수 있다. 연산 타입으로는 bf16, 단정도 실수(single precision, 혹은 float) 16비트를 사용하였다. 이는 가중치(weight) 값의 소수점 자릿수를 제한하여 효율적으로 메모리를 사용할 수 있다.

LoRA 랭크(rank)는 모델이 업데이트할 수 있는 저차원 가중치 행렬 크기를 의미하며, 이 값이 크면 모델이 더 많은 파라미터를 조정할 수 있지만, 그만큼 복잡도가 커질 수 있다. LoRA 알파(alpha) 값은 스케일링(scaling) 값으로 이 값이 클수록 저차원 가중치 행렬의 영향력이 더 커지고, 이 값이 낮으면 반대로 사전 학습 모델(기본 모델)의 가중치 행렬의 영향이 더욱 커진다. 본 연구에서는 각각 기본값인 8, 16을 사용하였다.

LoRA 드롭아웃(dropout)은 저차원 가중치 행렬의 과적합(overfitting)을 방지하기 위해 일부 가중치를 0으로 만드는 것이며, LoRA 비트 정밀도(bit precision)는 연산의 정밀도를 설정하는 것으로 보통 8비트 혹은 16비트를 사용한다. 이 값이 낮아질수록 연산 비용이 줄지만, 정밀도가 떨어질 수 있다. 본 연구에서는 각각 0, 8을 사용하였다. 그 외에 파라미터는 모두 LLaMa-Factory의 기본값을 사용하였다.

3-4 성능 평가 기준

본 연구에서는 평가 기준으로 전체 부동 소수점 연산(Total Floating Point Operations, 이하 Total FLOPs), 손실(loss) 값, 초당 샘플 처리 수(samples per seconds), 초당 처리 반복 수(steps per seconds)를 활용하였다.

Total FLOPs는 모델 학습 과정에서 수행한 부동 소수점 연산량으로 이 값이 크다면 학습 과정에서 많은 연산을 수행해 복잡성이 크다는 의미이고, 반대로 값이 낮다면 학습에 필요한 연산이 적다는 의미이다.

손실 값은 배치 크기 N, 최대 시퀀스 길이 T에 관하여 모델이 예측한 코드에 관한 확률 분포 $\hat{y}_{i,t}$와 실제 정답 코드 벡터 확률 분포 $y_{i,t}$간의 로그 우도(likelihood)를 의미한다. 이를 교차 엔트로피(cross entropy)라고 하며 이에 관한 수식은 아래 수식 1과 같다.

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{i=1}^{T} y_{i,t}\ln\left(\hat{y}_{i,t}\right) \quad (1)$$

Ⅳ. 실험 결과 및 고찰

4-1 실험 결과

위 장에서 기술한 동일한 하드웨어 환경과 학습 파라미터 환경에서 실험한 결과는 표 4와 같다.

표 4. 최종 실험 결과
Table 4. Final experiment results

	Qwen 2.5	LLaMA 3.2	Qwen 2.5 (Coder)	DeepSeek (Coder)
Total FLOPs	5.380	2.453	6.151	6.470
Loss	0.151	0.214	0.153	0.146
Runtime [seconds]	16740.183	5707.733	14773.92	16243.338
Sample per seconds	1.433	2.172	1.658	1.477
Steps per seconds	0.179	0.136	0.104	0.092

Total FLOPs의 경우 LLaMA 3.2에서 가장 낮은 결과를 보였는데, 이는 표 3에서 언급한 바와 같이 파라미터 수가 가장 적기 때문이다. 일반 Qwen 2.5 모델보다 Coder 모델이 더 높은 연산량을 보였으며 이는 일반화 모델보다 코딩에 특화된 모델이 더 많이 학습되었음을 알 수 있다.

손실 값에서는 DeepSeek 모델이 가장 성능이 높은 것으로 나타났고, Qwen 2.5에서 일반화 모델이 코딩에 특화된 모델보다 더 성능이 높은 것으로 나타났다. 각각의 모델에 관한 반복 횟수에 따른 손실 변화 그래프는 그림 3에서 6과 같다.

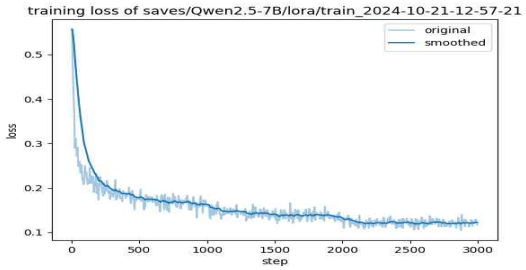

그림 3. Qwen 2.5 손실 그래프
Fig. 3. Qwen 2.5 loss graph

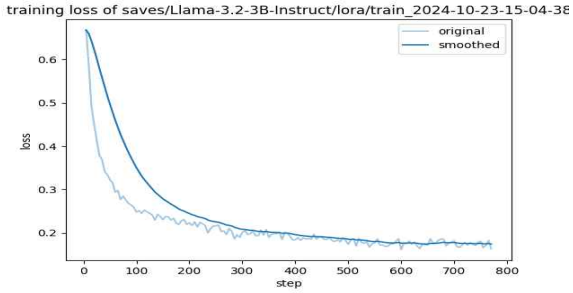

그림 4. LLaMA 3.2 손실 그래프
Fig. 4. LLaMA 3.2 loss graph

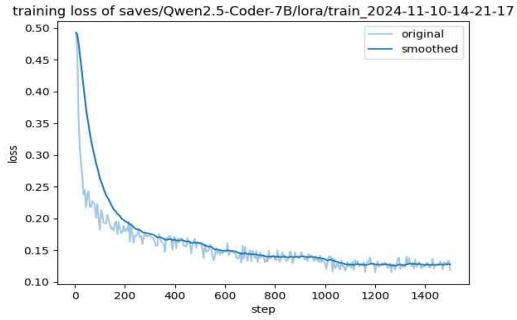

그림 5. Qwen 2.5 Coder 손실 그래프
Fig. 5. Qwen 2.5 Coder Loss Graph

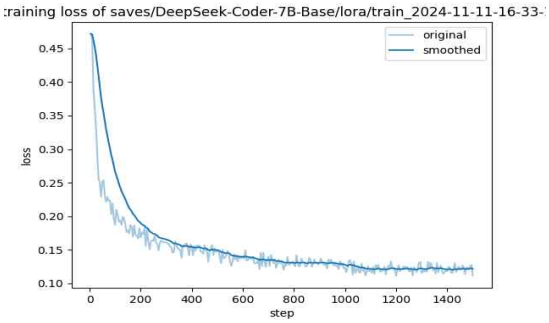

그림 6. DeepSeek Coder 손실 그래프
Fig. 6. DeepSeek Coder loss graph

위 4개의 손실 그래프의 개형을 봤을 때 LLaMA 3.2는 파라미터 수가 적어 800번의 단계만으로 훨씬 더 빨리 수렴하는 것을 볼 수 있고, Qwen 2.5 Coder와 DeepSeek Coder 모델이 1,500번의 단계만으로 더 빨리 수렴하는 것을 볼 수 있다. 이에 관한 내용은 다음 2절에서 자세히 논한다.

Total FLOPs를 봤을 때는 코딩에 특화된 모델이 학습 연산이 많아 일반화 모델보다 더 학습을 많이 한 것으로 보이지만, 성능에서는 일반화 모델이 아무래도 코딩에 특화된 모델보다 일반화 성능이 뛰어난 것으로 나타났다.

학습에 총 걸린 시간(runtime)과 초당 샘플 처리 수와 초당 처리 반복 수의 경우 Qwen 2.5가 다른 모델에 비해 더욱 오래 걸리는 것을 볼 수 있다.

4-2 고찰

앞서 그림 3에서 6의 손실 그래프 결과를 봤을 때 Qwen 2.5 일반 모델이 가장 단계가 크다. 이는 배치 크기와 관련되어 있다고 볼 수 있다. LLaMA 3.2와 같이 단계가 가장 적은 모델의 경우 하나의 반복 횟수만으로 더 많은 데이터를 처리했다는 의미이고 이는 결국 배치 크기가 크다는 의미이다. 모델의 학습 파라미터 수, QLoRA의 사용 여부, 모델의 복잡도 등에 따라 배치 크기가 커질 수도 있고 작아질 수도 있다는 의미이다. 이를 비교한 그래프는 그림 7과 같다.

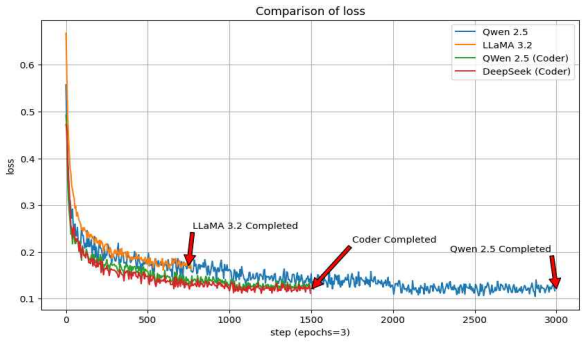

그림 7. 배치 크기에 따른 4가지 모델에 관한 학습 손실 비교 그래프
Fig. 7. Training loss comparison graph for four models according to batch size

다만 실제 테스트 환경에서 맥락과 관계없는 결과가 출력되거나 한국어로 물어봤을 때 중국어, 영어로 대답하는 경우가 있다. 이에 관한 결과는 그림 8과 같다.

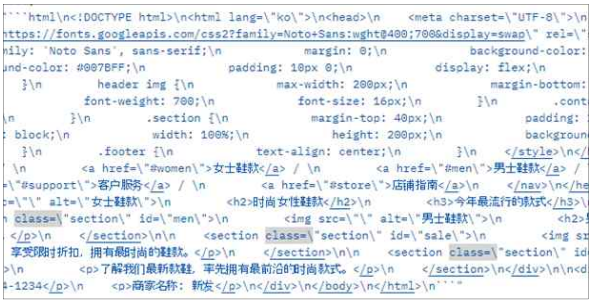

그림 8. 맥락과 관계없는 결과 예시
Fig. 8. Example results out of context

아울러 출력 결과인 코드가 길어지거나 이에 관한 설명이 길어지면 추론 시간이 길어지고 컴퓨팅 자원 사용량이 기하급수적으로 늘어나 이러한 점은 보완해야 할 것으로 사료된다. 또한, 복잡한 디자인을 요구하는 경우 이러한 문제가 더 심화하여 올바른 결과로 이어지지 않은 것으로 나타났다. 이에 관한 그림은 그림 9와 같다.

그림 9. 복잡한 디자인을 요구하는 경우 결과 예시
Fig. 9. Example results when complex design is required

그림 9와 같이, 레이아웃의 특정 요소 중 하나인 로고나 메뉴의 위치를 수정하라는 요청에 대해 문맥을 이해하지 못해 이를 반영하지 않았으며 중국어로 답변한다. 이러한 점에서 볼 때, 사용자 요구사항이 많아질수록 대처 능력이 떨어지는 것을 볼 수 있다.

본 연구에서는 웹 페이지를 디자인에 맞게 코딩해주는 것으로 일반적으로 코딩에 특화된 모델하고는 목적이 다르다는 한계가 있다. 여기서 일반적으로 코딩에 특화된 모델이란, 평가 방식과 학습에 활용된 프로그래밍 언어가 다르다는 의미이다.

먼저 평가 방식으로 프로그래밍 언어의 논리적 평가를 들 수 있다. 시간 복잡도(time complexity)와 공간 복잡도(space complexity)를 활용한 정량적 평가와 디자인에 관한 평가는 엄연히 다르다. 기존 코딩에 특화된 모델의 경우 기존 솔루션이 시간상 오래 걸리거나 메모리 점유율이 큰 코드의 경우 평가 점수를 낮게 준다.

마지막으로 HTML, CSS는 프로그래밍 언어가 아닌 마크업 언어(markup language)와 스타일 시트(style sheet)로써 작동되는 선언형 언어이며, JavaScript의 경우 프로그래밍 언어이긴 하지만 웹 페이지에서 보이는 이벤트를 담당하므로 사용자 친화적 디자인, 유려한 디자인이 담긴 코드 제작하고는 차이를 보일 수 있다. 특히, DeepSeek Coder에서 활용한 데이터셋의 경우 87개의 프로그래밍 언어가 활용되었는데 아래 표 5와 같이 상위 언어 비율과 표 6에서 HTML, CSS, JavaScript 비율과 큰 차이를 보인다[9]. 또한, 웹 페이지 구성에는 HTML, CSS, JavaScript가 상호작용하여 디자인을 구성하므로 이를 모두 활용하지 않으면 단순히 웹 페이지의 레이아웃만을 구성하는 결과를 보일 것으로 사료된다.

표 5. DeepSeek Coder 데이터셋 프로그래밍 언어 상위 5군 비율

Table 5. DeepSeek Coder dataset programming language top 5 clusters percentage

	Java	Python	C++	Type Script	PHP
Proportion[%]	18.63	15.12	11.39	7.60	7.38

표 6. DeepSeek Coder 데이터셋 웹 페이지 언어 비율

Table 6. DeepSeek Coder dataset web page language percentage

	HTML	CSS	JavaScript
Proportion[%]	3.77	0.71	6.75

Ⅴ. 결 론

본 논문에서는 2024년에 공개된 LLM 모델과 공개된 데이터셋을 활용해 웹 디자인에 적용할 수 있는 모델의 성능을 비교 분석하고, 이를 기반으로 비즈니스 모델에 활용할 가능성에 관해 연구하였다.

DeepSeek 모델의 경우 코딩에 특화된 모델로 가장 낮은 손실 값을 기록하며, 웹 디자인 작업에서 높은 성능을 보였으며 이는 프로젝트 레벨의 코딩 시나리오를 분류해 학습한 데이터셋과 풍부한 프로그래밍 언어를 활용한 기본 모델의 성능으로 판단된다. 그러나 웹 디자인에 필요한 HTML, CSS, JavaScript의 비율이 상대적으로 적어 웹 디자인에서 최적의 성능을 발휘하기 위해 추가적인 데이터셋 보완이 필요해 보인다.

Qwen 2.5 일반 모델의 경우 동종 코딩 특화 모델 대비 일반화 성능이 뛰어나 다양한 상황에서 유연성을 발휘할 수 있는 장점을 보였다. 그러나 학습과 추론 속도가 느리고 출력 결과가 길어질 때 메모리 사용량이 심하게 증가하고, 맥락과 관계없는 결과가 출력되는 것으로 보아 이에 관한 경량화, 최적화 연구가 추가로 필요해 보인다.

LLaMA 3.2의 경우 가장 경량화된 모델로써 가장 적은 학습 연산량을 기록했으며 가장 빠른 속도의 추론을 보였으나 손실 값이 비교적 높고, 학습 데이터셋이 한정적이기 때문에 부정확한 결과를 보였다.

본 연구에서 활용된 WebSight v1 데이터셋은 웹 페이지 구성 요소인 HTML, CSS, JavaScript 코드와 그에 관한 설명을 포함하고 있으며 웹 디자인 학습에 적합한 데이터셋이나 단순한 코드 생성 이상의 사용자 친화적 인터페이스를 얻기에는 한계를 보였다. 아울러 가장 좋은 결과를 보인 DeepSeek 모델과 함께 비추어 볼 때, 웹 디자인 언어 관련 비중이 작아 웹 디자인 작업에 최적화되지 않았다고 판단된다.

하지만 본 연구에서 LoRA와 QLoRA와 같은 미세조정 기법을 활용한 로컬 환경에서 학습 및 추론이 가능한 LLM을 통해 메모리 사용을 획기적으로 절감하면서도 효율적인 학습이 가능함을 보였다. 이를 통해 웹 디자인 모델 가능성에 관해 연구하고 각 모델의 성능과 한계를 비교 분석하였다. 이를 통해 비즈니스 환경에서 다양한 로컬 환경 위에서 웹 디자인 구현 모델을 적용할 수 있을 것으로 사료된다.

감사의 글

이 논문은 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원-지역지능화혁신인재양성사업의 지원을 받아 수행된 연구임(IITP-2025-RS-2022-00156334).

참고문헌

[1] M. Shao, A. Basit, R. Karri, and M. Shafique, “Survey of Different Large Language Model Architectures: Trends, Benchmarks, and Challenges,” *IEEE Access*, Vol. 12, pp. 188664-188706, 2024. https://doi.org/10.1109/ACCESS.2024.3482107

[2] J. Wang and Y. Chen, “A Review on Code Generation with LLMs: Application and Evaluation,” in *Proceedings of 2023*

IEEE International Conference on Medical Artificial Intelligence (MedAI), Beijing, China, pp. 284-289, November 2023. https://doi.org/10.1109/MedAI59581.2023.00044

[3] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, and Y. Zhang, "A Survey on Large Language Model (LLM) Security and Privacy: The Good, The Bad, and The Ugly," *High-Confidence Computing*, Vol. 4, No. 2, 100211, June 2024. https://doi.org/10.1016/j.hcc.2024.100211

[4] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, ... and G. Lample, "LLaMA: Open and Efficient Foundation Language Models," arXiv:2302.13971v1, February 2023. https://doi.org/10.48550/arXiv.2302.13971

[5] Meta. Llama 2: Open Foundation and Fine-Tuned Chat Models [Internet]. Available: https://ai.meta.com/research/publications/llama-2-open-foundation-and-fine-tuned-chat-models/.

[6] Meta. Llama 3.2: Revolutionizing Edge AI and Vision with Open, Customizable Models [Internet]. Available: https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/.

[7] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, ... and T. Zhu, "Qwen Technical Report," arXiv:2309.16609v1, September 2023. https://doi.org/10.48550/arXiv.2309.16609

[8] Qwen. Qwen2.5: A Party of Foundation Models! [Internet]. Available: https://qwenlm.github.io/blog/qwen2.5/.

[9] D. Guo, Q. Zhu, D. Yang, Z. Xie, K. Dong, W. Zhang, ... and W. Liang, "DeepSeek-Coder: When the Large Language Model Meets Programming - The Rise of Code Intelligence," arXiv:2401.14196v2, January 2024. https://doi.org/10.48550/arXiv.2401.14196

[10] N. Houlsby, A. Giurgiu, S. Jastrzębski, B. Morrone, Q. de Laroussilhe, A. Gesmundo, ... and S. Gelly, "Parameter-Efficient Transfer Learning for NLP," in *Proceedings of the 36th International Conference on Machine Learning*, Long Beach: CA, pp. 2790-2799, June 2019.

[11] E. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, ... and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," in *Proceedings of 2022 International Conference on Learning Representations (ICLR 2022)*, Online, April 2022. https://doi.org/10.48550/arXiv.2106.09685

[12] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient Finetuning of Quantized LLMs," in *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS 2023)*, New Orleans: LA, pp. 10088-10115, December 2023. https://doi.org/10.48550/arXiv.2305.14314

[13] H. Laurençon, L. Tronchon, and V. Sanh, "Unlocking the Conversion of Web Screenshots into HTML Code with the WebSight Dataset," arXiv:2403.09029v1, March 2024. https://doi.org/10.48550/arXiv.2403.09029

이태준(Tae-Jun Lee)

2020년 : 배재대학교 (공학사-컴퓨터공학)
2022년 : 배재대학교 대학원 (공학석사-컴퓨터공학)
2022년~현 재: 배재대학교 대학원 컴퓨터공학 박사과정
※관심분야 : 딥러닝(Deep learning), 머신러닝(Machine learning), 빅데이터(Big data) 등

박정석(Jung-Seok Park)

1997년 : 동의대학교 (공학사-전산통계학)
2001년 : 부산대학교 대학원 (공학석사-전자계산학)
2001년~2008년: ㈜아이티즌 책임연구원
2008년~2015년: ㈜한백전자 부장
2025년~현 재: 배재대학교 스마트ICT융합전공 박사과정
2016년~현 재: ㈜바인드소프트 대표
※관심분야 : AI IOT, ROS(Robot Operationg System), Embedded System 등

한 개(Kai Han)

2008년 : 배재대학교 (학사-전자상거래학)
2012년 : 배재대학교 대학원 (석사-무역학과)
2017년 : 배재대학교 대학원 (박사-컴퓨터공학)
2021년~현 재: 우송대학교 융합경영학과, 조교수
※관심분야 : 멀티미디어 문서 처리, 디지털 도서관, 정보 처리, 웹 서비스 등

정회경(Hoe-Kyung Jung)

1985년 : 광운대학교 (공학사-컴퓨터공학)
1987년 : 광운대학교 대학원 (공학석사-컴퓨터공학)
1993년 : 광운대학교 대학원 (공학박사-컴퓨터공학)
1994년~현 재: 배재대학교 컴퓨터공학과 교수
※관심분야 : 머신러닝(Machine learning), 빅데이터(Big data), 임베디드 시스템(Embedded system), IoT 등