

# GROKING: 과적합을 넘어서는 일반화 - 소규모 알고리즘 데이터셋에 대한 TING

알레시아 파워, 유리 부르다, 해리 에드워즈, 이고르 바부쉬킨  
오픈AI

베던트 미스라\*  
Google

## 추상적인

본 논문에서는 작은 알고리즘으로 생성된 데이터 세트에 대한 신경망의 일반화를 연구할 것을 제안합니다. 이 설정에서는 데이터 효율성, 암기, 일반화 및 학습 속도에 대한 질문을 매우 자세하게 연구할 수 있습니다. 어떤 상황에서는 신경망이 데이터의 패턴을 "그로킹 (grokking)"하는 과정을 통해 학습하여 무작위 확률 수준에서 완벽한 일반화까지 일반화 성능을 향상시키고 이러한 일반화 개선이 과적합 지점을 훨씬 지나서 발생할 수 있음을 보여줍니다. 또한 데이터셋 크기에 따른 일반화를 연구하고 데이터셋이 작을 수록 일반화를 위해 더 많은 양의 최적화가 필요하다는 사실을 발견했습니다. 우리는 이러한 데이터 세트가 딥 러닝의 잘 이해되지 않은 측면, 즉 유한한 훈련 데이터 세트의 암기를 넘어 과도하게 매개변수화된 신경망의 일반화를 연구하기 위한 비옥한 기반을 제공한다고 주장합니다.

## 1. 소개

과도하게 매개변수화된 신경망의 일반화는 고전 학습 이론에서 파생된 직관을 무시하기 때문에 오랫동안 기계 학습 커뮤니티의 관심 대상이었습니다. 이 논문에서 우리는 작은 알고리즘으로 생성된 데이터셋에 대한 훈련 네트워크가 훈련 세트의 성능과 명확하게 분리된 비정상적인 일반화 패턴을 안정적으로 나타낼 수 있음을 보여줍니다. 이러한 효과는 자연 데이터에서 파생된 데이터셋에 나타나는 효과보다 훨씬 더 두드러집니다 (그림 1 참조). 예를 들어 왼쪽). 이러한 실험은 단일 GPU에서 빠르게 재현될 수 있으므로 일반화 이론을 위한 편리한 테스트베드가 됩니다.

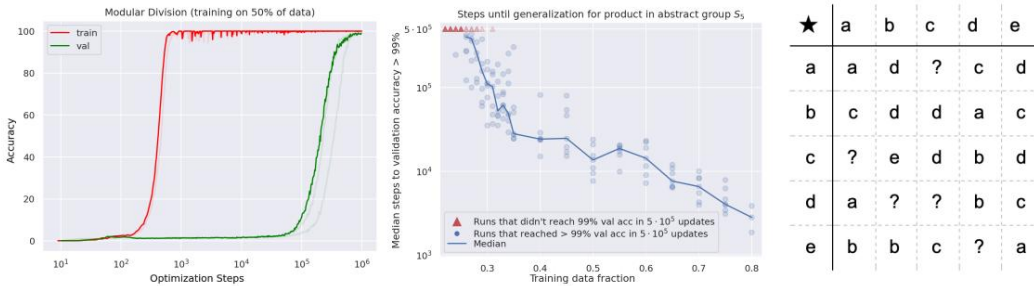


그림 1: 왼쪽. Grokking: 알고리즘 데이터 세트에 과적합한 후 일반화의 극적인 예입니다. 훈련 세트에 있는 데이터의 50%를 사용하여 나눗셈 mod 97의 이진 연산을 훈련합니다.

97개의 잔기 각각은 오른쪽 그림의 표현과 유사하게 별도의 기호로 네트워크에 표시됩니다. 빨간색 곡선은 훈련 정확도를 나타내고 녹색 곡선은 검증 정확도를 나타냅니다. 훈련 정확도는 103개 미만의 최적화 단계에서 완벽에 가까워지지만 검증 정확도가 해당 수준에 도달하려면 거의 106 단계가 필요하며 105 단계까지는 일반화에 대한 증거가 거의 없습니다. 센터. 99% 검증 정확도에 도달하는 데 필요한 훈련 시간은 훈련 데이터 비율이 감소함에 따라 빠르게 증가합니다. 오른쪽. 작은 이진 연산 테이블의 예입니다. 독자가 어떤 요소가 누락되었는지 추측해 보시기 바랍니다.

\*Vedant는 이 작업 당시 OpenAI에 있었습니다.

arXiv:201.02177v1

우리가 고려하는 데이터 세트는  $a \circ b = c$  형식의 이진 연산 테이블입니다. 여기서  $a, b, c$  는 내부 구조가 없는 이진 기호이고  $\circ$  는 이진 연산입니다. 이항 연산의 예로는 덧셈, 순열 구성, 이변량 다항식이 있습니다. 가능한 모든 방정식의 적절한 하위 집합에 대해 신경망을 훈련하는 것은 스토쿠 퍼즐을 푸는 것과 마찬가지로 이진 연산 테이블의 공백을 채우는 것과 같습니다. 그림 1의 오른쪽에 예가 나와 있습니다. 방정식에 포함된 모든 고유한 요소  $a, b, c$  에 대해 고유한 추상 기호를 사용하기 때문에 네트워크는 요소의 내부 구조를 인식하지 못하며 다음 사항에 대해 학습해야 합니다. 다른 요소와의 상호 작용을 통해서만 속성을 얻을 수 있습니다. 예를 들어 네트워크에서는 십진수 표기법의 숫자나 줄 표기법의 순열을 볼 수 없습니다.

우리의 기여는 다음과 같습니다:

- 우리는 신경망이 다양한 방식으로 빈 슬롯을 일반화할 수 있음을 보여줍니다. 바이너리 연산 테이블.
- 우리는 심각한 과적합 이후 오랜 시간이 지나서 때때로 검증 정확도가 우연 수준에서 완벽한 일반화를 향해 갑자기 증가하기 시작한다는 것을 보여줍니다. 우리는 이런 현상을 '그로킹'이라고 부릅니다. 그림 1에 예가 나와 있습니다. • 다양한 이진 연산에 대한 데이터 효율성 곡선을 제시합니다. • 일반화에 필요한 최적화의 양이 빠르게 증가한다는 것을 경험적으로 보여줍니다.

데이터 세트 크기가 감소함에 따라 증가합니다.

- 다양한 최적화 세부 사항을 비교하여 데이터 효율성에 미치는 영향을 측정합니다. 우리는 체중 감소가 우리가 연구하는 작업에 대한 일반화를 개선하는 데 특히 효과적이라는 것을 발견했습니다. • 우리는 이러한 네트워크에서 학습된 기호 임베딩을 시각화하고 때때로 기호로 표시되는 수학적 개체의 인식 가능한 구조를 발견합니다.

## 2 방법

우리의 모든 실험에서는  $a \circ b = c$  형식의 방정식 데이터 세트에 대해 훈련된 작은 변환기를 사용했습니다. 여기서 "a", "o", "b", "=" 및 "c" 는 각각 별도의 토큰입니다. 연구된 작업, 아키텍처, 훈련 하이퍼파라미터 및 토큰화에 대한 세부 정보는 부록 A.1에서 확인할 수 있습니다.

## 3 가지 실험

### 3.1 과적합을 넘어서는 일반화

딥 러닝 실무자는 검증 손실 감소가 멈춘 후 검증 정확도가 약간 향상되는 것을 확인하는 데 익숙합니다. 일부 상황에서는 검증 손실의 이중 하강이 문서화되었지만 Nakkiran 등 실무자들 사이에서는 이례적인 것으로 간주됩니다. (2019); Belkin et al. (2018); d'Ascoli et al. (2020). 우리가 연구하는 소규모 알고리즘 데이터 세트에서는 다양한 모델, 최적화 도구 및 데이터 세트 크기에 대해 초기 과적합 후 향상된 일반화가 발생하며 경우에 따라 이러한 효과가 극도로 두드러집니다. 그림 1에는 모듈식 분할에 대한 일반적인 예가 나와 있습니다. 여기서 우리는 훈련 정확도가 최적이 가까워지는 데 필요한 것보다 1000배 더 많은 최적화 단계 후에만 검증 정확도가 우연 수준 이상으로 증가하기 시작한다는 것을 알 수 있습니다. 그림 4에는 훈련/검증 손실도 표시되어 있으며 검증 손실의 이중 하강을 볼 수 있습니다.

우리는 이러한 동작이 할당된 최적화 예산 내에서 네트워크가 일반화한 최소 데이터 세트 크기에 가까운 데이터 세트 크기에 대한 모든 이진 작업에서 일반적이라는 것을 발견했습니다. 데이터세트 크기가 클수록 학습 곡선과 검증 곡선은 서로를 더 밀접하게 추적하는 경향이 있습니다.

#### 3.1.1 학습 시간 곡선

일반적인 지도 학습 문제에서 훈련 데이터의 양을 줄이면 최적화 절차가 훈련 데이터를 보간할 수 있는 경우 모델의 수렴 일반화 성능이 감소합니다. 우리 설정에서는 다른 현상을 관찰했습니다. 수렴된 성능은 훈련 데이터 세트 크기 범위 내에서 100%로 일정하게 유지되지만, 데이터 세트 크기가 감소함에 따라 해당 성능을 달성하는 데 필요한 최적화 시간이 빠르게 증가합니다.

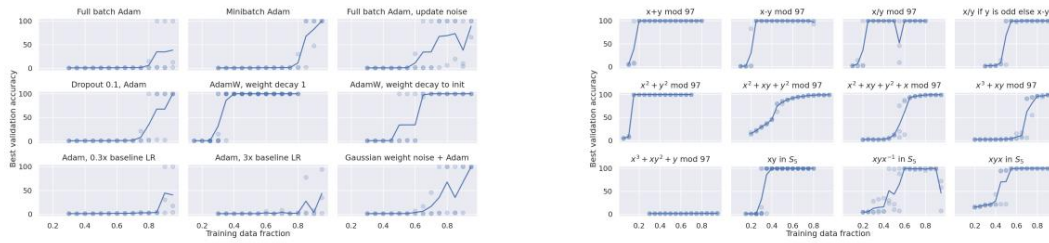


그림 2: 왼쪽. 서로 다른 최적화 알고리즘은 추상 그룹 S5의 제곱 학습 문제에 대해 105 단계의 최적화 예산 내에서 서로 다른 일반화 양으로 이어집니다.

가중치 감소는 일반화를 가장 크게 향상시키지만 전체 배치 최적화 프로그램과 모델을 사용하더라도 훈련 데이터의 높은 비율에서 가중치 또는 활성화 노이즈가 없는 경우에도 일부 일반화가 발생합니다. 최적 이 아닌 선택 하이퍼파라미터는 일반화를 심각하게 제한합니다. 표시되지 않음: 모든 최적화 방법에 대해 103~104 번 업데이트한 후 훈련 정확도가 100%에 도달합니다. 오른쪽. 다양한 알고리즘 데이터 세트에서 105 단계를 수행 한 후 최고의 검증 정확도를 얻었으며 평균 3개의 시드가 있습니다. 일반화는 직관적으로 더 복잡하고 덜 대칭적인 작업을 위해 더 높은 비율의 데이터에서 발생합니다.

그림 1(가운데)은 추상 그룹 S5의 제곱에 대해 검증 성능이 처음 99%에 도달할 때까지 최적화 단계의 중앙값을 보여줍니다. 데이터의 25~30% 부근에서 훈련 데이터가 1% 감소하면 일반화에 소요되는 평균 시간이 40~50% 증가합니다. 검증 정확도 > 99%까지의 단계 수는 데이터 세트 크기가 감소함에 따라 빠르게 증가하지만, 학습 정확도가 처음 99%에 도달할 때까지의 단계 수는 일반적으로 데이터 세트 크기가 감소하고 103~104 최적화 단계 범위 에 머물면서 감소하는 경향을 보입니다. 우리는 네트워크를 일반화할 수 있는 모든 알고리즘 작업에서 데이터 세트 크기가 감소함에 따라 일반화에 도달할 때까지 최적화 시간이 기하급수적으로 증가하는 유사한 패턴을 관찰했습니다.

### 3.2 다양한 문제에 대한 불평

우리는 부록 A.1.1에 나열된 다양한 이진 연산에 대해 사용 가능한 모든 방식의 다양한 부분으로 구성된 교육 데이터 세트에 대해 세 번의 실행에 걸쳐 평균 정확도를 측정했습니다. 결과는 그림 2(오른쪽)에 나와 있습니다.

피연산자는 관련되지 않은 추상 기호로 신경망에 표시되므로 소수  $p$  및  $0$ 이 아닌  $x, y$ 를 갖는 연산  $x+y \pmod{p-1}$  및  $x*y \pmod{p}$ 는 신경망과 구별할 수 없습니다. 네트워크의 관점(그리고 유사하게  $x^2 \pmod{p-1}$  및  $x/y \pmod{p}$ ). 이는  $0$ 이 아닌 모든 잔여물이 소수 모듈로(modulo a prime)로 기본 근의 거듭제곱으로 표현될 수 있기 때문입니다.

이 표현은 모듈로  $p-1$ 의 모듈러 덧셈 과 모듈로  $p$ 의 모듈러 곱셈의 동등성(기호 이름 변경까지)을 보여줍니다. 그림 2(오른쪽)에서  $x^2 \pmod{p-1}$ 와  $x/y \pmod{p}$ 는 일반화가 발생하기 위해 실제로 거의 동일한 양의 데이터를 사용한다는 것을 알 수 있습니다.

그림 2(오른쪽)에 나열된 연산 중 일부는 피연산자 순서  $(x+y, x^2 * y, x^2)$ 를 기준으로 대칭입니다. 이러한 연산은 밀접하게 관련된 비대칭 연산  $(x^2 \pmod{p-1}, x/y, x+x)$ 보다 일반화에 더 적은 데이터가 필요한 경우가 많습니다.  $x^2 + xy + y^2$  우리는 이 효과가 부분적으로 아키텍처에 종속적일 수 있다고 믿습니다. 왜냐하면 변환기가 위치 임베딩을 무시하여 피연산자의 대칭 기능을 학습하는 것이 쉽기 때문입니다.

의 데이터 비율에서 최적화 예산을 허용했습니  $x^2 + xy^2 + y \pmod{97}$ 는 일부 작업 내에서 일반화로 이어지지 않았습니다(예:  $x$ 는 최대 95% 다. 수렴된 모델은 실제 패턴을 찾지 않고 교육 데이터 세트를 효과적으로 기억했습니다. 이러한 모델에서는 데이터가 사실상 무작위입니다.

$y$ 가 홀수인 경우  $x/y \pmod{p}$ , 그렇지 않으면  $x^2 \pmod{p}$  연산은 네트워크가 여러 간단한 연산의 혼합을 학습하도록 요구합니다. 특히  $x$ 의 역할은 다음에서 잔여물로 해석되어야 합니다. 짝수  $y$ 와 짝을 이룰 때 덧셈 그룹, 홀수  $y$ 와 짝을 이룰 때 곱셈 그룹의 나머지로 사용됩니다. 이는 그룹 또는 링 작업을 통해 명확하게 해석할 수 없는 작업에 대해서도 일반화가 발생할 수 있음을 보여줍니다.

### 3.3 절제와 트릭

우리는 네트워크가 데이터 세트에서 더 나은 일반화를 유도할 수 있는 방법을 알아보기 위해 다양한 형태의 정규화를 시도했습니다. 여기에서는 전체 배치 경사하강법, 확률적 경사하강법, 크거나 작은 학습률, 다양한 개입을 위한 특정 데이터 세트 S5에 대한 데이터 효율성 곡선을 제시합니다.

잔류 탈락 Srivastava et al. (2014), 체중 감쇠 Loshchilov & Hutter (2017) 및 경사 노이즈 Neelakantan et al. (2015). 결과는 그림 2(왼쪽)에 나와 있습니다.

우리는 가중치 감소를 추가하면 대부분의 다른 개입에 비해 필요한 샘플 양이 절반 이상으로 데이터 효율성에 매우 큰 영향을 미친다는 사실을 발견했습니다. 우리는 네트워크 초기화를 향한 가중치 감소도 효과적이지만 원점을 향한 가중치 감소만큼 효과적이지는 않다는 것을 발견했습니다. 이는 대략 0의 가중치가 작은 알고리즘 작업에 적합하다는 사전 설명이 가중치 감쇠의 우수한 성능을 일부 설명하지만 전부는 아니라고 믿게 만듭니다. 최적화 프로세스에 약간의 노이즈(예: 미니배치 사용으로 인한 그래디언트 노이즈, 그래디언트 계산 전후에 가중치에 적용되는 가우스 노이즈)를 추가하는 것은 일반화에 도움이 되며, 이러한 노이즈가 더 나은 일반화를 위한 더 평평한 최소값을 찾기 위해 최적화를 유도할 수 있다는 생각과 일치합니다.

우리는 일반화가 일어나기 위해서는 상대적으로 좁은 범위(1자릿수 이내)에서 학습률을 조정해야 한다는 것을 발견했습니다.

### 3.4 임베딩의 질적 시각화

일반화되는 네트워크에 대한 통찰력을 얻기 위해 모듈 추가 및 S5의 경우 출력 레이어의 매트릭스를 시각화했습니다. 그림 3에서는 행 벡터의 t-SNE 플롯을 보여줍니다.

일부 네트워크의 경우 플롯에서 기본 수학적 객체의 구조가 명확하게 반영되어 있습니다. 예를 들어, 모듈식 추가의 원형 토폴로지는 각 요소에 8을 더하여 형성된 '수선'으로 표시됩니다. 구조는 가중치 감소로 최적화된 네트워크에서 더욱 분명해집니다.

## 4 토론

우리는 우리가 연구한 데이터 세트에서 작은 알고리즘 이진 연산 테이블, 이중 하강 또는 후기 일반화와 같은 효과, 체중 감소와 같은 개입으로 인한 일반화 개선이 눈에 띌 수 있다는 것을 확인했습니다. 이는 이러한 데이터 세트가 일반화 측면을 조사하기에 좋은 장소가 될 수 있음을 시사합니다. 예를 들어, 우리는 제안된 다양한 최소 평탄도 측정값이 우리 환경의 일반화와 상관관계가 있는지 테스트할 계획입니다.

우리는 또한 이러한 신경망의 임베딩 공간을 시각화하면 자연스러운 종류의 구조를 보여줄 수 있다는 것을 확인했습니다. 예를 들어 모듈러 산술 문제에서 임베딩 토폴로지는 원이나 원통이 되는 경향이 있습니다. 또한 네트워크가 다양한 잔여물을 기준으로 임베딩을 특이하게 구성하는 경향이 있음을 알 수 있습니다. 이러한 수학적 개체의 속성은 우리에게 친숙하지만, 그러한 시각화가 언젠가는 새로운 수학적 개체에 대한 직관을 얻는 데 유용한 방법이 될 수 있다고 추측합니다.

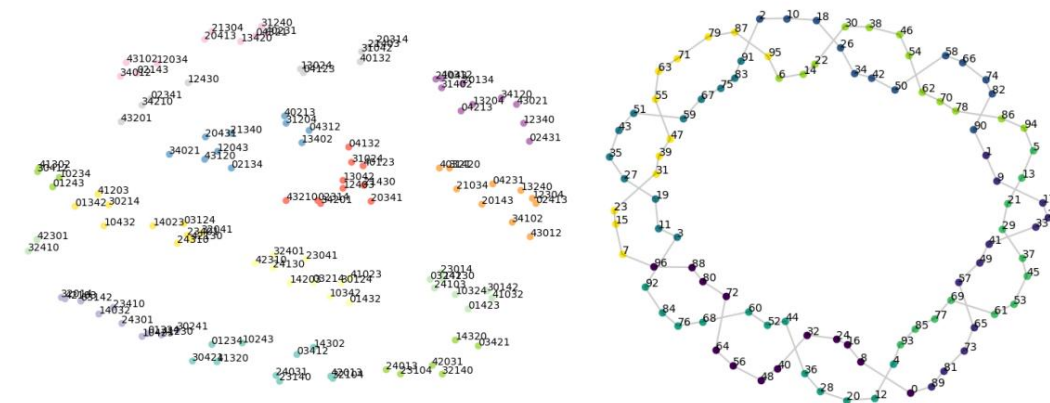


그림 3: 왼쪽. S5에서 훈련된 네트워크의 출력 레이어 가중치에 대한 t-SNE 투영입니다. 순열의 클러스터를 볼 수 있으며, 각 클러스터는 하위 그룹 (0, 3) (1, 4), (1, 2), (3, 4) 또는 해당 켈레 중 하나의 coset입니다. 오른쪽. 모듈 추가에 대해 훈련된 네트워크의 출력 레이어 가중치에 대한 t-SNE 투영입니다. 선은 각 요소에 8을 더한 결과를 보여줍니다. 색상은 각 요소 모듈로 8의 나머지를 표시합니다.

또한 훈련 데이터 세트의 크기를 줄이면 주어진 성능 수준에 도달하는 데 필요한 최적화 단계 수가 빠르게 증가한다는 흥미로운 현상을 문서화했습니다 . 이는 더 적은 양의 데이터에 대한 성능을 위해 계산을 교환하는 방법을 나타내기 때문에 향후 작업에서 다른 데이터 세트에도 효과가 있는지 조사하는 것이 유용할 것입니다.

## 참고자료

Mikhail Belkin, Daniel Hsu, Siyuan Ma, Soumik Mandal. 현대 머신러닝의 조화 연습과 편향-분산 트레이드오프. arXiv 사전 인쇄 arXiv:1812.11118, 2018.

스테판 다스콜리, 레벤트 사군, 줄리오 비롤리. 삼중 하강과 두 종류의 과적합: 어디에 & 왜 나타나는가? arXiv 사전 인쇄 arXiv:2006.03509, 2020.

모스타파 데가니(Mostafa Dehghani), 스테판 가우우스(Stephan Gouws), 오리올 비날스(Oriol Vinyals), 야콥 우슈코레이트(Jakob Uszkoreit), 루카스 카이저( ukasz Kaiser). 만능인 변압기. arXiv 사전 인쇄 arXiv:1807.03819, 2018.

알렉스 그레이브스. 순환 신경망의 적응형 계산 시간. arXiv 사전 인쇄 arXiv:1603.08983, 2016.

알렉스 그레이브스, 그렉 웨인, 이보 다니헬카. 신경 튜링 기계. arXiv 사전 인쇄 arXiv:1410.5401, 2014.

에드워드 그레펜스테트, 칼 모리츠 헤르만, 무스타파 솔레이만, 필 블런섬. 무한한 기억으로 변환하는 법을 배웁니다. 신경 정보 처리 시스템의 발전, 28: 1828-1836, 2015.

..

제프 호흐라이터(Sepp Hochreiter)와 JURGEN 슈미트후버(Jurgen Schmidhuber). 플랫 최소값. 신경 계산, 9(1):1-42, 1997.

이딩 장, 베남 네이사부르, 호세인 모바히, 달림 크리스난, 사미 벤지오. 환상적인 일반화 조치와 이를 찾을 수 있는 곳. arXiv 사전 인쇄 arXiv:1912.02178, 2019.

ukasz Kaiser와 Ilya Sutskever. 신경 GPU 학습 알고리즘. arXiv 사전 인쇄 arXiv:1511.08228, 2015.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy 및 Ping Tak Peter Tang. 딥 러닝을 위한 대규모 배치 훈련: 일반화 격차와 날카로운 최소값. CoRR, abs/1609.04836, 2016. URL <http://arxiv.org/abs/1609.04836>.

일리아 로슈일로프(Ilya Loschilov)와 프랭크 후터(Frank Hutter). 분리된 체중 감소 정규화. arXiv 사전 인쇄 arXiv:1711.05101, 2017.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak 및 Ilya Sutskever. 심층 이중 하강: 더 큰 모델과 더 많은 데이터가 해를 끼치는 곳. arXiv 사전 인쇄 arXiv:1912.02292, 2019.

Arvind Neelakantan, Luke Vilnis, Quoc V Le, Ilya Sutskever, Lukasz Kaiser, Karol Kurach 및 James Martens. 경사 노이즈를 추가하면 매우 깊은 네트워크에 대한 학습이 향상됩니다. arXiv 사전 인쇄 arXiv:1511.06807, 2015.

스캇 리드와 난도 데 프레이타스. 신경 프로그래머-통역사. arXiv 사전 인쇄 arXiv:1511.06279, 2015.

David Saxton, Edward Grefenstette, Felix Hill 및 Pushmeet Kohli. 수학적 분석 신경 모델의 추론 능력. arXiv 사전 인쇄 arXiv:1904.01557, 2019.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever 및 Ruslan Salakhutdinov. 드롭아웃: 신경망의 과적합을 방지하는 간단한 방법입니다. 기계 학습 연구 저널, 15(1):1929-1958, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser 및 Illia Polosukhin. 주의가 필요한 전부입니다. arXiv 사전 인쇄 arXiv:1706.03762, 2017.

제이슨 웨스턴, 슈미트 초프라, 앙투안 보르데스. 메모리 네트워크. arXiv 사전 인쇄 arXiv:1410.3916, 2014.



- 각 매개변수의 업데이트 방향에 전체 배치 및 가우스 노이즈가 추가된 Adam 최적화 프로그램 ( $W \leftarrow W + lr \cdot (\Delta W + \epsilon)$ ), 여기서 가우시안 단위에서 샘플링됨,  $\Delta W$  는 표준 Adam 가중치 업데이트이고  $lr$  은 학습률) • 잔여 드롭아웃 0.1이 추가된 모델의 Adam 최적화 프로그램 • 가중치 감

최소 10이 포함된 AdamW 최적화 프로그램(대부분의 다른 실험에서는 기본

설정) • 원점 대신 초기화 방향으로 가중치 감쇠 10이 포함된 AdamW 최적화 프로그램 • 학습률 3이 포

함된 Adam 최적화 프로그램 10<sup>-4</sup> • 학습률 3 · 10<sup>-3</sup> 의 Adam 옵티마이저 • 표준 편차 0.01의 가우

스 가중치 노이즈를 갖는 모델의 Adam 옵티마이저

(즉, 모델에서 각 매개변수  $W$ 가  $W + 0.01 \cdot \epsilon$  로 대체되

고 가우스 단위에서 샘플링됨).

섹션 3.1.1에 보고된 실험의 경우 일반화를 완료하는 데 걸리는 시간 증가를 더 잘 포착하기 위해 최적화 예산을 5 · 10<sup>5</sup> 최적화 단계로 늘렸습니다.

섹션 3.1에 보고된 실험에 대해 우리는 최적화 프로세스가 얼마나 늦게 일반화를 시작할 수 있는지 강조하기, 그리고 사용 위해 가중치 감소 없이 최적화 예산을 10<sup>6</sup> Adam 최적화 프로그램으로 늘렸습니다.

우리는 7개의 무작위 시드에 대한 결과를 집계한 섹션 3.1.1의 실험을 제외하고 3개의 무작위 시드를 사용하여 각 데이터 세트 크기에 대해 각 실험을 반복했습니다.

## A.2 추가 그림

그림 4에서는 그림 1의 정확도 곡선에 해당하는 손실 곡선을 보여줍니다.

그림 5에서는 네트워크가 실제로 해결할 수 있는 이진 연산 테이블의 예를 보여줍니다.

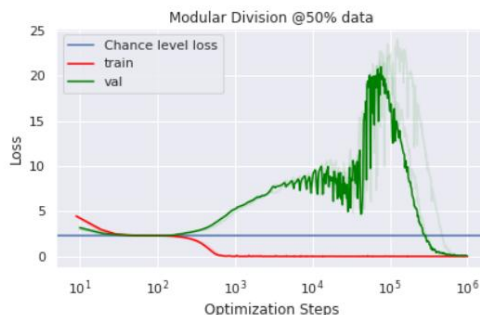


그림 4: 모듈 분할, 학습 및 검증에 대한 손실 곡선. 두 번째 하강을 시작하기 전에 검증 손실이 10<sup>2</sup> 단계에서 약 10<sup>5</sup> 단계의 최적화 단계로 증가하는 것을 볼 수 있습니다.

## A.3 관련 작업

이 논문에서는 작고 간단한 알고리즘 데이터셋에 대한 훈련 및 일반화 역학을 연구합니다. 과거에는 알고리즘 데이터 세트를 사용하여 기호 및 알고리즘 추론을 수행하는 신경망의 기능을 조사했습니다. 예를 들어 무작위로 생성된 시퀀스를 복사, 역전, 정렬하고 여러 자리 숫자의 산술 연산을 수행하는 작업은 시퀀스 간 모델 Graves et al.의 표준 벤치마크로 사용되었습니다. (2014), 웨스트턴 외. (2014) Kaiser & Sutskever (2015) Reed & De Freitas (2015), Grefenstette 외. (2015), Zaremba & Sutskever (2015), Graves (2016), Dehghani et al. (2018). 그러나 일반적으로 이러한 작업에서는 입력 시퀀스 길이와 관련하여 일반화가 종종 연구되는 무제한 데이터 체제에서의 성능에 중점을 둡니다. 일부 논문에서는 알고리즘 작업 Reed & De Freitas(2015)의 샘플 복잡성을 연구하지만 대부분 아키텍처 선택의 영향에 중점을 둡니다. 대조적으로 우리는 아키텍처에 구애받지 않는다고 믿는 현상에 중점을 두고 데이터가 제한된 체제에서 일반화 현상을 연구합니다.





그림 5: 네트워크가 완벽하게 채울 수 있는 네트워크에 제시된 이진 연산 테이블 중 하나. 각 기호는 독자의 편의를 위해 영어, 히브리어 또는 그리스 알파벳의 문자로 표시됩니다. 여기에 어떤 작업이 표시되는지 독자가 추측해 보시기 바랍니다.

bAbi Weston et al.과 같은 알고리즘으로 생성된 추론 데이터세트. (2015)은 데이터 제한 체제에서 일반화 연구 작업을 장려합니다. 그러나 이러한 데이터 세트에 대한 대부분의 결과는 특정 아키텍처 또는 훈련 기술의 성능에 대한 점 추정에 초점을 맞추는 반면, 우리의 주요 관심은 특정 아키텍처가 훈련 데이터를 완전히 기억할 수 있는 지점을 지나 일반화의 변화를 지적하는 것입니다.

Neelakantan et al. (2015)은 알고리즘 작업에 대해 "grok와 같은" 학습 곡선을 가지고 있지만 이는 최적화 난이도와 관련이 있는 반면, 우리의 현상은 특히 일반화에 관한 것입니다.

Saxton et al. (2019) 그들은 산술 및 미분과 같이 절차적으로 생성된 수학 문제에 대한 일반화를 연구 하지만 대부분의 경우 이러한 작업은 우리가 연구한 단순한 이진 연산 문제보다 더 복잡하므로 우리가 연구하는 현상의 종류를 관찰하는 데 적합하지 않습니다. 마스터하려면 매우 많은 수의 샘플이 필요하기 때문에 이 문서에서 설명합니다.

Jiang et al. (2019) 그들은 컨볼루션 신경망에 대한 수많은 일반화 또는 복잡성 측정을 연구하여 어떤 것이 일반화 성능을 예측하는지 확인했습니다.

그들은 매개변수 선택에 대한 훈련된 신경망의 민감도를 정량화하는 것을 목표로 하는 평판도 기반 측정이 가장 예측 가능성이 높다는 것을 발견했습니다. 우리는 그로킹 현상이 일어날 것이라고 추측했습니다.



우리는 이 연구에서 더 나은 일반화를 위해 최적화를 추진하는 SGD의 노이즈로 인한 것일 수 있다고 보고 하며 향후 작업에서 이러한 조치가 그로킹을 예측하는지 여부를 조사하기를 희망합니다.

Zhang et al. (2016)은 딥 러닝에 일반적으로 사용되는 크기의 신경망이 임의의 훈련 데이터를 보간할 수 있으며 적절한 최적화 절차를 사용하여 의미상 의미 있는 레이블로 훈련할 때 일반화할 수 있음을 발견했습니다. 우리의 작업은 신경망이 일반화 없이 작은 알고리즘 훈련 데이터 세트를 보간할 수 있지만 SGD로 더 오랫동안 훈련할 때 일반화를 시작하는 관련 현상을 보여줍니다.

Nakkiran et al. (2019); Belkin et al. (2018)은 모델 및 최적화 절차 용량의 함수로서 손실의 이중 하강 현상에 중점을 둡니다. 그들은 전통적인 U자형 검증 손실 곡선이 일부 설정(신경망 훈련 포함)에서 훈련 데이터를 보간하는 데 필요한 최소 용량 주위에서 시작하는 손실의 두 번째 하강을 따른다는 것을 발견했습니다.

우리는 일부 실험에서 훈련량 의 함수로 검증 손실(정확성은 아니지만)의 두 번째 하강을 관찰했으며 이는 훈련 데이터를 보간하는 지점을 지나서 발생합니다.

우리는 우리가 설명하는 현상이 Nakkiran et al.에서 설명된 이중 하강 현상과 다를 수 있다고 믿습니다. (2019); Belkin et al. (2018) 왜냐하면 우리는 훈련 손실이 처음으로 매우 작아졌을 때(일부 실험에서는 수만 에포크 ) 훨씬 이후 손실의 두 번째 하강을 관찰하고 정확도의 비단조적인 동작을 관찰하지 않기 때문입니다. 우리가 연구하는 작은 알고리즘 데이터 세트의 설정은 Nakkiran et al.에서 연구된 자연 데이터 세트보다 미묘한 일반화 현상을 연구하기 위한 더 작고 다루기 쉬운 놀이터를 제공합니다. (2019).

#### A.4 여러 이상값을 기억하여 일반화

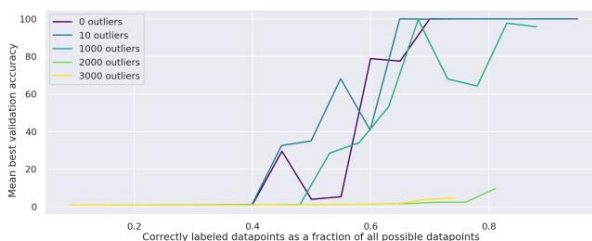


그림 6: 훈련 데이터에  $k \in [0, 10, 100, 1000, 2000, 3000]$  이상치(무작위 라벨이 있는 예)를 도입할 때 데이터 효율성에 미치는 영향. 이상치 수가 적으면 일반화 성능에 눈에 띄게 영향을 주지 않지만, 수가 많으면 일반화 성능이 크게 저하됩니다.

이 섹션에서는 훈련 데이터세트에  $k$  개의 이상치를 도입하여 이전 작업의 수정된 버전에 대한 데이터 효율성 곡선을 보여줍니다. 보다 정확하게는 실험 시작 시 훈련 세트에서  $k$ 개의 방정식을 무작위로 샘플링 하고 그 답을 훈련 데이터에서 무작위로 샘플링된 다른  $k$  개의 방정식에 대한 답으로 바꿉니다. 훈련 데이터의 나머지 방정식과 검증 데이터의 모든 방정식은 이전과 동일하게 유지됩니다.

이 상황에서는 다음 시나리오 중 하나가 전개되는 것을 상상할 수 있습니다. 이전과 같이 최적화되고 정규화된 신경망의 모델 클래스가 이러한 "잡음이 있는" 데이터 세트를 보간할 만큼 충분히 크지 않은 경우, 잘 일반화되지만 훈련 데이터의 잡음을 제거하는 솔루션으로 수렴하는 절차를 상상할 수 있습니다 (즉,  $c = a \circ b$  예측). 이상치 방정식  $a, b \rightarrow c$  ( $c = c$ )에 대한 답이기도 합니다. 다른 극단적인 경우는 최적화 절차가 데이터를 보간하는 네트워크를 찾을 수 있지만 결과 모델이 일반화되지 않을 수도 있습니다. 왜냐하면 이전보다 훨씬 더 복잡한 함수(간단한 함수 +  $k$  예외가 인코딩됨)를 나타내야 하기 때문입니다. 훈련 데이터).

실험에서 우리는 첫 번째 움푹이 발생하지 않는다는 것을 발견했습니다. 모든 실험은 어떤 시점에서 100% 훈련 정확도에 도달하고 이 시점은 이상값  $k$ 의 수를 변경해도 크게 영향을 받지 않습니다. 두 번째 현상은 훈련 데이터 백분율 범위와 이상치 수  $k$ 에서 발생합니다.  $k$ 가 증가하면 최적화 절차가 일반화되는 모델로 수렴되는 훈련 데이터 백분율 범위가 감소합니다. 그러나 소수의 이상값(최대 1000개)을 도입하는 효과는 그리 뚜렷하지 않습니다. 그림 6을 참조하세요. 우리는 이를 네트워크 및 최적화 절차의 용량이 필요한 용량을 훨씬 초과한다는 추가 증거로 해석합니다.

