12. Distilling the Knowledge in a Neural Network

앙상블에 포함된 지식을 하나의 단일 모델로 압축해 배포하기 쉽게 만들자!

Introduction

훈련할 때는 크고 복잡한 모델을 써도 괜찮지만, 배포(deployment) 단계에서는 **지연·속도·메모리 제한** 때문에 작은 모델이 필요하다. 하지만 작은 모델을 직접 학습시킨 모델은 큰 모델에 비해 성능이 떨어짐.

→ 큰 모델(또는 앙상블)의 지식을 작은 모델에 전이해보자

→ **Distillation**

: 큰 모델이 예측한 **soft targets**를 작은 모델의 훈련 목표로 사용

- soft targets는 정답뿐 아니라 오답 간 상대적 확률, 클래스 간 미세한 구조를 담고 있어 작은 모델이 큰 모델의 일반화 패턴을 학습하는 데 매우 효과적이다.

- **Temperature Softmax**

  logits를 부드럽게 만들기 위해 softmax에 높은 temperature(T) 를 적용

  T ↑ → 분포가 고르게 퍼짐 → 클래스 간 관계가 더 잘 드러남

  작은 모델은 이 soft한 분포를 학습하면서 큰 모델의 지식을 흡수한다.

- soft targets는 정답은 아니지만 비슷한 클래스에 적당한 확률을 주고, 따라서 클래스 간 구조를 효과적으로 전달할 수 있고 따라서 hard label만 학습하는 것보다 더 많은 정보를 담을 수 있다.

Distillation

- Softmax 출력층

$$q_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$$

- 가장 단순한 형태 - transfer set에 대해 큰 모델이 높은 temperature의 softmax로 생성한 soft target을 사용하여 작은 모델(distilled model)을 훈련.
  - 작은 모델을 훈련할 때는 이 높은 temperature를 사용하지만, 훈련이 끝난 후에는 temperature를 1로 한다.
  - 가장 좋은 방법은 서로 다른 두 목적 함수의 가중 평균을 사용하는 것.
    1. soft targets와의 cross-entropy
       - 큰 모델의 soft targets 생성에 사용된 것과 동일한 높은 temperature를 작은 모델의 softmax에도 적용하여 계산
    2. 정답 레이블과의 cross-entropy
       - temperature = 1을 사용

    → 실험 결과, 두 번째 목적 함수의 가중치를 훨씬 낮게 주는 것이 가장 좋은 성능을 보임
       - soft target이 생성하는 gradient의 크기는 1 / T^2 비율로 작아지므로, hard target과 soft target을 함께 사용할 경우 gradient를 T^2 로 곱해주는 것이 중요하다. 이렇게 하면 temperature를 조절해도 두 손실의 상대적 기여도가 유지된다.

Matching logits is a special case of distillation

transfer set의 각 sample은 작은 모델의 logit z_i 에 대해서 cross-entropy gradient dC / dz_i 를 제공

- 큰 모델의 logits v_i , soft target 확률 p_i 를 생성한다고 할 때 gradient:

$$\frac{\partial C}{\partial z_i} = \frac{1}{T}(q_i - p_i) = \frac{1}{T}\left(\frac{e^{z_i/T}}{\sum_j e^{z_j/T}} - \frac{e^{v_i/T}}{\sum_j e^{v_j/T}}\right)$$

  - 만약 temperature 가 logits 의 크기보다 훨씬 크다면:

$$\frac{\partial C}{\partial z_i} \approx \frac{1}{T}\left(\frac{1+z_i/T}{N+\sum_j z_j/T} - \frac{1+v_i/T}{N+\sum_j v_j/T}\right)$$

logits가 transfer case마다 zero-mean 되도록

$$\sum_j z_j = \sum_j v_j = 0$$

를 가정하면 위 식은 단순화되어:

$$\frac{\partial C}{\partial z_i} \approx \frac{1}{NT^2}(z_i - v_i)$$

→ high-temperature limit에서는 distillation이 사실상

$$\frac{1}{2}(z_i - v_i)^2$$

를 **최소화하는것**과 동일하다.

Preliminary experiments on MNIST

MNIST 실험 결과

- Distillation은 작은 모델의 성능을 크게 끌어올림
  - 큰 모델: 67 errors
  - 작은 모델(정규화 없음): 146 errors
  - 작은 모델 + soft targets 학습: 74 errors

    → 작은 모델이 큰 모델의 일반화 능력을 상당 부분 전이 받음.
- Temperature는 작은 모델의 크기에 따라 최적값이 달라짐

    - 모델이 클 때(≥300 units) → T ≥ 8이면 거의 동일
    - 모델이 매우 작을 때(≈30 units) → T = 2.5~4가 최적

      → 작은 모델일수록 너무 크거나 작은 T는 오히려 해가 됨.
- Transfer set에서 특정 클래스(예: '3')를 완전히 제거해도 generalization 가능
    - '3'을 훈련에서 한 번도 보지 않아도
    - bias만 조정하면 test '3'의 **98.6%**를 정확히 인식

      → Distillation은 "클래스 간 구조적 정보"를 전이시키기 때문에, unseen class도 어느 정도 인식 가능.
- Label bias 조정만으로 성능이 크게 향상됨
    - mnist 7/8만 보고도 전체 테스트 오류를 47.3% → 13.2%로 감소

      → Teacher model의 soft target이 클래스 간 상대적 관계를 잘 전달하기 때문.

Experiments on speech recognition

자동 음성 인식(ASR)에 사용되는 심층 신경망(DNN) 음향 모델에 대해 앙상블링의 효과를 조사

→ distillation 전략 사용 시, 동일한 훈련 데이터를 사용해 개별적으로 학습된 여러 모델의 앙상블 대신, **단일 모델에 앙상블의 효과를 증류하는 결과를 낳으며**, 동일한 크기의 모델을 직접 학습한 경우보다 훨씬 더 좋은 성능을 발휘한다.

- 최신 ASR 시스템:

  일반적으로 DNN을 사용하여 (짧은) 시간적 컨텍스트의 특징을 파형에서 추출하고, 이를 HMM(Hidden Markov Model)의 이산 상태에 대한 확률 분포로 매핑함
    - DNN을 훈련할 때 언어 모델의 영향을 고려하기 위해 모든 가능한 경로에 대해 주변화(marginalization)하는 것이 가능(그리고 바람직)하지만,
    - 일반적으로는 강제 정렬(forced alignment)된 정답 HMM 상태에 대해 frame-by-frame 분류를 수행하도록 cross entropy를 최소화하여 훈련한다.

$$\theta = \arg\max_{\theta'} P(h_t \mid s_t; \theta')$$

실험

- 동일한 아키텍처와 훈련 절차를 사용하여 서로 다른 초기 파라미터 값으로 10개의 모델을 훈련
- distillation에는 온도 **1, 2, 5, 10**을 실험했으며, hard targets의 cross entropy에는 가중치 0.5 사용

  → 10개 모델 앙상블을 사용할 때 얻어지는 frame classification accuracy 개선의 **80% 이상이 distilled model로 성공적으로 전이되었다**.

결과

System	Test Frame Accuracy	WER
Baseline	58.9%	10.9%
10xEnsemble	61.1%	10.7%
Distilled Single model	60.8%	10.7%

→ distillation된 단일 모델이 soft targets를 생성한 10개 모델 앙상블의 평균 결과와 거의 비슷한 성능을 낸다.

Training ensembles of specialists on very big datasets

앙상블에 대한 또 다른 반론 = 개별 모델이 큰 신경망이고 데이터셋이 매우 큰 경우, 비록 병렬화가 쉽더라도 훈련 시 필요한 계산량이 너무 많아진다.

→ **클래스 중 서로 혼동되기 쉬운(subset) 부분만을 각 specialist 모델이 담당하도록 하는 전략으로 앙상블 학습에 필요한 총 계산량을 줄일 수 있음을 보여줌**

The JFT dataset

Google 내부 데이터셋으로, 1억 개의 라벨링된 이미지와 15,000개의 라벨을 포함. JFT baseline 모델은 6개월 동안 다수의 코어에서 **비동기적 스토캐스틱 경사하강 (asynchronous SGD)** 으로 학습된 대규모 CNN

사용된 병렬화

- **데이터 병렬(data parallelism)**:

    - 여러 replica가 서로 다른 minibatch를 처리하며, 각 replica는 자신의 gradient를 공유된 parameter server로 전송
    - parameter server는 gradient를 모아 업데이트된 파라미터를 다시 replicas로 전송
- **모델 병렬(model parallelism)**:
    - 거대한 네트워크를 여러 코어로 분할하여 각 subset의 뉴런을 각 코어에 배치

→ 앙상블 학습은 이 두 종류의 병렬화 위에 또 하나의 병렬 계층(third type of parallelism) 을 더 얹는 것과 같다.

JFT 1: Tea party; Easter; Bridal shower; Baby shower; Easter Bunny; ...
JFT 2: Bridge; Cable-stayed bridge; Suspension bridge; Viaduct; Chimney; ...
JFT 3: Toyota Corolla E100; Opel Signum; Opel Astra; Mazda Familia; ...

Specialist Models

특정 specialist model은 자신이 신경 쓰지 않는 클래스들은 모두 하나의 “dustbin class”로 합쳐 softmax 규모를 크게 줄일 수 있다.

Specialist 학습 방법

- Specialist는 generalist model의 가중치로 초기화
- 자신의 domain에서 온 예시 50%, 일반 training set에서 random sampling된 50%로 학습
- Specialist는 oversampled된 클래스가 있기 때문에, dustbin class에 대한 예측 logit을 보정해 줘야 한다.

Assigning classes to specialists

specialist를 위한 object category 그룹을 만들기 위해, generalist model이 자주 혼동하는 클래스들에 집중

- 정답 라벨을 사용하지 않기 위해, generalist model의 예측 covariance matrix를 사용하여 clustering을 적용

  절차

    - covariance matrix의 column에 대해 online K-means 알고리즘 적용

- Table 2와 같은 reasonable clusters 생성
- 여러 알고리즘을 시도했지만 유사한 결과 도출됨

Performing inference with ensembles of specialists

specialist 모델을 distillation하기 전에, specialist를 포함하는 앙상블이 얼마나 잘 작동하는지 먼저 평가

1. generalist model이 예측한 top-n 클래스를 선택 (n=1)
2. 그 top-1 클래스와 교집합을 갖는 specialist 모델들로 active set A_k 구성.
3. 다음 KL-divergence를 최소화하는 전체 분포 q 계산

$$KL(p^g, q) + \sum_{m \in A_k} KL(p^m, q)$$

Result

훈련 완료된 generalist model에서 시작하여, specialist 모델들은 매우 빠르게 훈련시킬 수 있다.(몇 일)-?

- 각 specialist는 독립적으로 학습 가능

System	Conditional Test Accuracy	Test Accuracy
Baseline	43.1%	25.0%
+ 61 Specialist models	45.9%	26.1%

→ Specialist 추가로 top-1 accuracy 4.4%p 증가

# of specialists covering	# of test examples	delta in top1 correct	relative accuracy change
0	350037	0	0.0%
1	141993	+1421	+3.4%
2	67161	+1572	+7.4%
3	38801	+1124	+8.8%
4	26298	+835	+10.5%
5	16474	+561	+11.1%
6	10682	+362	+11.3%
7	7376	+232	+12.8%
8	4703	+182	+13.6%
9	4706	+208	+16.6%
10 or more	9082	+324	+14.1%

Specialist가 커버하는 경우 top-1 accuracy 변화

→ Specialist가 많이 관여할수록 정확도 개선 폭이 커지고, Specialist 모델들이 병렬학습이 가능하므로 규모 확장이 쉬움

Soft Targets as Regularizers

단일 하드 타겟으로는 결코 담을 수 없는 많은 유용한 정보를 소프트 타겟은 담을 수 있음

System & training set	Train Frame Accuracy	Test Frame Accuracy
Baseline (100% of training set)	63.4%	58.9%
Baseline (3% of training set)	67.3%	44.5%
Soft Targets (3% of training set)	65.4%	57.0%

→ 데이터의 단 3% (약 2,000만 개 예시)만을 사용해 baseline 모델을 하드 타겟으로 학습하면 심각한 과적합이 발생한다.

→ 동일한 모델을 소프트 타겟으로 학습하면 전체 학습 데이터에서 얻을 수 있는 정보의 거의 대부분을 회복할 수 있었음. 조기 종료도 없이.

⇒ **소프트 타겟이 전체 데이터를 사용해 학습된 모델이 발견한 규칙성을 다른 모델에게 전달하는 데 매우 효과적이다.**

Using soft targets to prevent specialists from overfitting

JFT 데이터셋 실험에서 사용한 specialist 모델들은 자신이 담당하지 않는 모든 클래스를 단일 dustbin 클래스에 넣음

→ specialist는 자신의 special class들이 매우 많이 포함된 데이터로 학습되는데, 따라서 실질적인 훈련 데이터셋 크기는 매우 작으며, 그 special class들에 **심하게 과적합**되기 쉬움.

이를 해결하기 위한 방법으로 soft target 을 이용할 수 있음.

- specialist 모델을 generalist 모델의 가중치로 초기화한
- 그 specialist에게 non-special classes에 대한 soft targets를 함께 제공하여 학습시키면,
- specialist는 non-special 클래스에 대한 지식을 거의 그대로 유지할 수 있다.

Relationship to Mixtures of Experts

데이터의 하위 집합에 대해 학습된 specialist를 사용하는 것은 mixture of experts와 어느 정도 유사함

- Mixture of experts: 게이팅 네트워크(gating network)를 사용하여 각 예시를 어떤 expert에게 할당할지 결정하는 확률을 계산
  - experts는 자신에게 할당된 예시를 처리하도록 학습
  - 게이팅 네트워크는 각 예시에 대해 어떤 expert를 사용할지 학습
  - 문제점
    1. 각 expert가 보는 weighted training set이 모든 다른 expert에 따라 계속 변함
    2. 게이팅 네트워크가 동일한 예시에 대해 여러 expert의 성능을 비교하고 그에 따라 확률을 수정해야 함

    → mixture of experts는 실제로 가장 유용할 만한 상황—즉, 엄청난 규모의 데이터셋에서 클래스 간 명확한 subset 구조가 있는 경우—에서도 거의 사용되지 않는다.
- specialist 여러 개를 병렬로 학습하는 것 이 훨씬 낫다.