NLP Challenge : Trial and Error

Team : State_Of_The_Art
Dongju Park

School of Electrical Engineering and Computer Science
Gwangju Institute of Science and Technology (GIST)
Meta-Evolutionary Machine Intelligence Laboratory (MEMI lab)

NLP Challenge

NLP Challenge

TensorFlow Korea

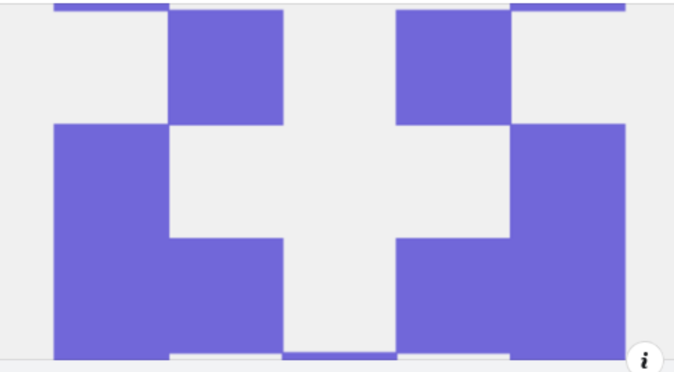
님이 그룹에 링크를 공유했습니다: TensorFlow KR.
11월 19일 ·

https://github.com/naver/nlp-challenge

GITHUB.COM
naver/nlp-challenge
NLP Shared tasks (NER, SRL) using NSML. Contribute to naver/nlp-challenge development by creating an account on GitHub.

님이 그룹에 링크를 공유했습니다: TensorFlow KR.
11월 20일 ·

이미 광고된 것 같은데 추가로 첨언해서 복붙합니다.
네이버 서치 앤 클로바 조직 NLP/대화 Inho Kang 리더님 曰
" 우수한 품질 내는 팀, NLP/대화 팀 입사 지원시 공식 코딩테스트 면제 등 특혜드립니다."
" 팀 TO 아직 여유 많으니 관심있으신 분들은 많은 지원 바랍니다."

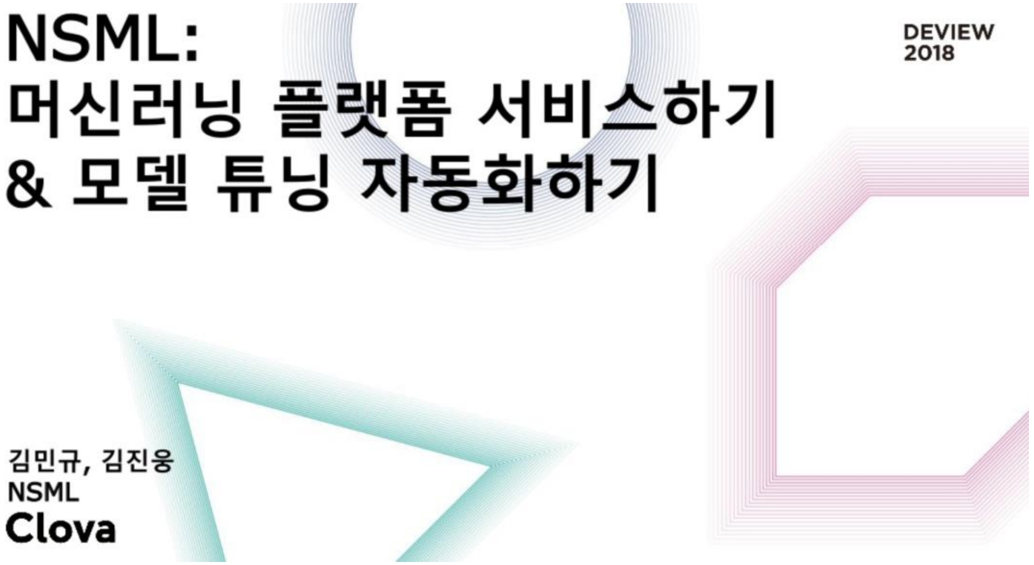


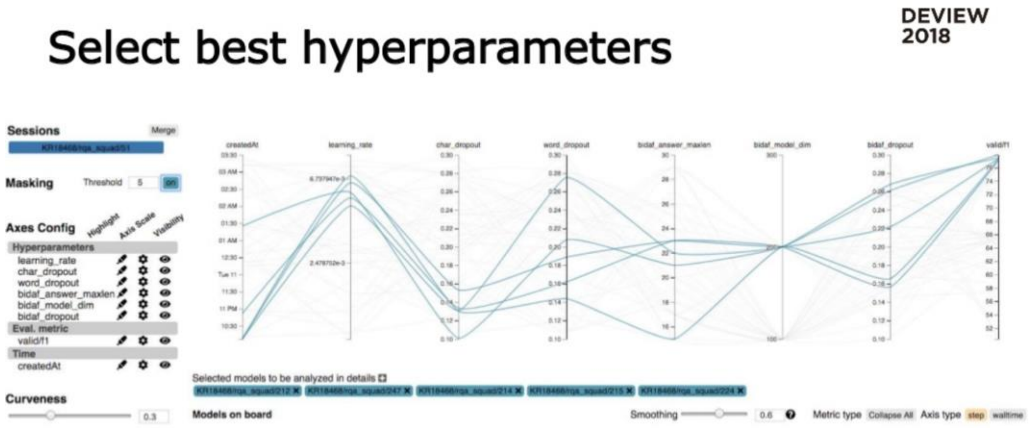
GITHUB.COM
naver/nlp-challenge
NLP Shared tasks (NER, SRL) using NSML. Contribute to naver/nlp-

NLP Challenge

FINAL + LAB + PERSONAL

NLP Challenge





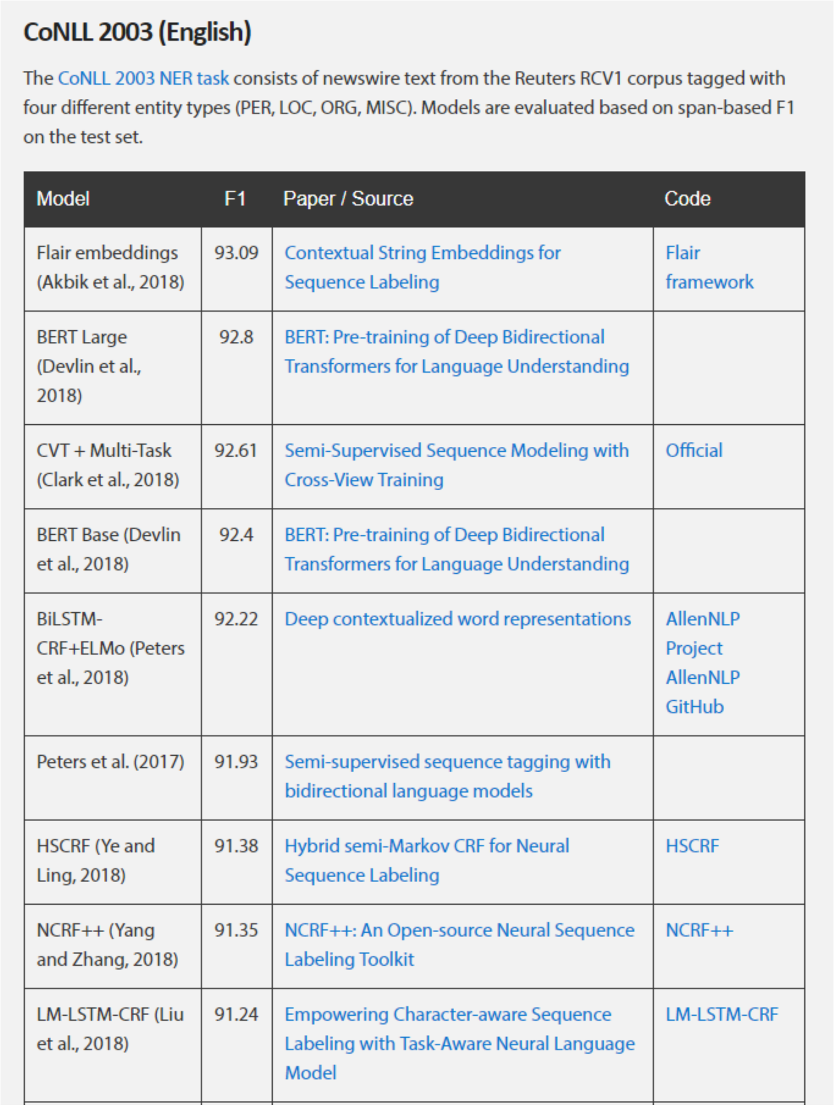
Model

CoNLL 2003 (English)

The CoNLL 2003 NER task consists of newswire text from the Reuters RCV1 corpus tagged with four different entity types (PER, LOC, ORG, MISC). Models are evaluated based on span-based F1 on the test set.

Model	F1	Paper / Source	Code
Flair embeddings (Akbik et al., 2018)	93.09	Contextual String Embeddings for Sequence Labeling	Flair framework
BERT Large (Devlin et al., 2018)	92.8	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	
CVT + Multi-Task (Clark et al., 2018)	92.61	Semi-Supervised Sequence Modeling with Cross-View Training	Official
BERT Base (Devlin et al., 2018)	92.4	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	
BiLSTM-CRF+ELMo (Peters et al., 2018)	92.22	Deep contextualized word representations	AllenNLP Project AllenNLP GitHub
Peters et al. (2017)	91.93	Semi-supervised sequence tagging with bidirectional language models	
HSCRF (Ye and Ling, 2018)	91.38	Hybrid semi-Markov CRF for Neural Sequence Labeling	HSCRF
NCRF++ (Yang and Zhang, 2018)	91.35	NCRF++: An Open-source Neural Sequence Labeling Toolkit	NCRF++
LM-LSTM-CRF (Liu et al., 2018)	91.24	Empowering Character-aware Sequence Labeling with Task-Aware Neural Language Model	LM-LSTM-CRF

Model



CoNLL 2003 (English)

The CoNLL 2003 NER task consists of newswire text from the Reuters RCV1 corpus tagged with four different entity types (PER, LOC, ORG, MISC). Models are evaluated based on span-based F1 on the test set.

Model	F1	Paper / Source	Code
Flair embeddings (Akbik et al., 2018)	93.09	Contextual String Embeddings for Sequence Labeling	Flair framework
BERT Large (Devlin et al., 2018)	92.8	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	
CVT + Multi-Task (Clark et al., 2018)	92.61	Semi-Supervised Sequence Modeling with Cross-View Training	Official
BERT Base (Devlin et al., 2018)	92.4	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	
BiLSTM-CRF+ELMo (Peters et al., 2018)	92.22	Deep contextualized word representations	AllenNLP Project AllenNLP GitHub
Peters et al. (2017)	91.93	Semi-supervised sequence tagging with bidirectional language models	
HSCRF (Ye and Ling, 2018)	91.38	Hybrid semi-Markov CRF for Neural Sequence Labeling	HSCRF
NCRF++ (Yang and Zhang, 2018)	91.35	NCRF++: An Open-source Neural Sequence Labeling Toolkit	NCRF++
LM-LSTM-CRF (Liu et al., 2018)	91.24	Empowering Character-aware Sequence Labeling with Task-Aware Neural Language Model	LM-LSTM-CRF

알고있고

간단하고

쉽게 만들 수 있는

BILSTM-CRF + (ELMo)

Model

CoNLL 2003 (English)

The CoNLL 2003 NER task consists of newswire text from the Reuters RCV1 corpus tagged with four different entity types (PER, LOC, ORG, MISC). Models are evaluated based on span-based F1 on the test set.

Model	F1	Paper / Source	Code
Flair embeddings (Akbik et al., 2018)	93.09	Contextual String Embeddings for Sequence Labeling	Flair framework
BERT Large (Devlin et al., 2018)	92.8	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	
CVT + Multi-Task (Clark et al., 2018)	92.61	Semi-Supervised Sequence Modeling with Cross-View Training	Official
BERT Base (Devlin et al., 2018)	92.4	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	
BiLSTM-CRF+ELMo (Peters et al., 2018)	92.22	Deep contextualized word representations	AllenNLP Project AllenNLP GitHub
Peters et al. (2017)	91.93	Semi-supervised sequence tagging with bidirectional language models	
HSCRF (Ye and Ling, 2018)	91.38	Hybrid semi-Markov CRF for Neural Sequence Labeling	HSCRF
NCRF++ (Yang and Zhang, 2018)	91.35	NCRF++: An Open-source Neural Sequence Labeling Toolkit	NCRF++
LM-LSTM-CRF (Liu et al., 2018)	91.24	Empowering Character-aware Sequence Labeling with Task-Aware Neural Language Model	LM-LSTM-CRF

NER Model Baseline for NSML

NER baseline Model 구조

- Bidirectional RNN + CRF
- 어절, 음절(RNN) Concat 하여 사용

```
import tensorflow as tf

class Model:
    def __init__(self, parameter):
        self.parameter = parameter

    def build_model(self):
        self._build_placeholder()

        # { "morph": 0, "morph_tag": 1, "tag" : 2, "character": 3, .. }
        self._embedding_matrix = []
        for item in self.parameter["embedding"]:
            self._embedding_matrix.append(self._build_embedding(item[1], item[2], name="embedding_" + item[0]))

        # 각각의 임베딩 값을 가져온다
        self._embeddings = []
        self._embeddings.append(tf.nn.embedding_lookup(self._embedding_matrix[0], self.morph))
        self._embeddings.append(tf.nn.embedding_lookup(self._embedding_matrix[1], self.character))

        # 음절을 이용한 임베딩 값을 구한다.
        character_embedding = tf.reshape(self._embeddings[1], [-1, self.parameter["word_length"], self.parameter["embedding"][1][2]])
        char_len = tf.reshape(self.character_len, [-1])
```

Model

NER Model Baseline for NSML

NER baseline Model 구조

- Bidirectional RNN + CRF
- 어절, 음절(RNN) Concat 하여 사용

baseline	2018-11-19 21:05:54	모델번호: 1 *팀명: nsmlteam*	88.0977

Hyper-parameters

Epochs	40
Batch size	20
Learning rate	0.01
Word embedding size	32
Char embedding size	32
Tag embedding size	32
Lstm units	32
Char lstm units	64

Model

NER Model Baseline for NSML

NER baseline Model 구조

- Bidirectional RNN + CRF
- 어절, 음절(RNN) Concat 하여 사용

baseline	2018-11-19 21:05:54	모델번호: 1 팀명: nsmlteam	88.0977

88.0977

Hyper-parameters

Epochs	40
Batch size	20
Learning rate	0.01
Word embedding size	32
Char embedding size	32
Tag embedding size	32
Lstm units	32
Char lstm units	64

49.4154

Model

baseline	2018-11-19 21:05:54	모델번호:1 팀명: nsmlteam	88.0977

Hyper-parameters

Epochs	40
Batch size	20
Learning rate	0.01
Word embedding size	32
Char embedding size	32
Tag embedding size	32
Lstm units	32
Char lstm units	64

49.4154

Hyper-parameters

Epochs	20
Batch size	10
Learning rate	0.02
Word embedding size	16
Char embedding size	16
Tag embedding size	16
Lstm units	16
Char lstm units	32

66.3427

Model

baseline	2018-11-19 21:05:54	모델번호: 1 팀명: nsmlteam	88.0977

Hyper-parameters

Epochs	20
Batch size	10
Learning rate	0.02
Word embedding size	16
Char embedding size	16
Tag embedding size	16
Lstm units	16
Char lstm units	32

66.3427

Hyper-parameters

Epochs	100
Batch size	1000
Learning rate	0.02
Word embedding size	16
Char embedding size	16
Tag embedding size	16
Lstm units	16
Char lstm units	32

71.5061

Model

Tuning
Tuning Tuning
Submit Submit
Submit Submit
Submit
Submit Tuning
Submit Submit
Submit
Tuning
Submit Submit Tuning
Submit
Tuning Submit
Submit
Submit
Tuning Tuning

Model

Tuning Tuning Submit Tuning
Submit Submit
Submit Submit Submit
Submit Submit Tuning Submit
Tuning Submit
Submit Submit Submit Tuning
Tuning Submit Submit
Tuning Submit Tuning

```
nsml.report(summary=True, scope=locals(), train__loss=avg_cost, step=epoch)
nsml.save(epoch)
```

모델번호 = epoch

baseline	2018-11-19 21:05:54	모델번호: 1 팀명: nsmlteam	88.0977

Model

main.py

```
parser.add_argument('--train_lines', type=int, default=50, required=False, help='Maximum train lines')
```

dataset_batch.py

```
def get_data_batch_size(self, n, train=True):
    if train:
        for i, step in enumerate(range(0, self.parameter["train_lines"], n)):
            if len(self.morphs[step:step + n]) == n:
                yield self.morphs[step:step+n], self.ne_dicts[step:step+n], self.characters[step:step+n], \
                    self.sequence_lengths[step:step+n], self.character_lengths[step:step+n], \
                    self.labels[step:step+n], i
    else:
        for i, step in enumerate(range(0, self.parameter["train_lines"], n)):
            if len(self.morphs[step:step+n]) == n:
                yield self.morphs[step:step+n], self.ne_dicts[step:step+n], self.characters[step:step+n], \
                    self.sequence_lengths[step:step+n], self.character_lengths[step:step+n], \
                    self.labels[step:step+n], i
```

Model

baseline	2018-11-19 21:05:54	모델번호: 1 팀명: nsmlteam	88.0977

Hyper-parameters

Epochs	100
Batch size	1000
Learning rate	0.02
Word embedding size	16
Char embedding size	16
Tag embedding size	16
Lstm units	16
Char lstm units	32

75 epochs : 71.5061

Hyper-parameters

Train lines	90000
Epochs	20
Batch size	64
Learning rate	0.02
Word embedding size	16
Char embedding size	16
Tag embedding size	16
Lstm units	16
Char lstm units	32

2 epochs : 85.5220

Model

baseline	2018-11-19 21:05:54	모델번호: 1 팀명: nsmlteam	88.0977

keep_prob word_length batch_size

embedding_size lstm_units

Tuning
Tuning Submit Tuning
Submit
Submit Submit
Submit
Submit Submit Tuning Submit
Tuning Submit
Submit Submit Tuning
Submit
Tuning Submit
Submit
Submit
Tuning Tuning

Model

baseline	2018-11-19 21:05:54	모델번호:1 팀명: nsmlteam	88.0977

Hyper-parameters

Train lines	90000
Epochs	20
Batch size	64
Learning rate	0.02
Word embedding size	16
Char embedding size	16
Tag embedding size	16
Lstm units	16
Char lstm units	32

85.5220

Hyper-parameters

Train lines	90000
Epochs	20
Batch size	128
Learning rate	0.005
Word embedding size	128
Char embedding size	128
Tag embedding size	128
Lstm units	128
Char lstm units	128

88.1035

Model

Hyper-parameters

Train lines	90000
Epochs	20
Batch size	128
Learning rate	0.005
Word embedding size	128
Char embedding size	128
Tag embedding size	128
Lstm units	128
Char lstm units	128

88.1035

Hyper-parameters tuning

training data score

↓

test data score

Model

Hyper-parameters

Train lines	90000
Epochs	20
Batch size	128
Learning rate	0.005
Word embedding size	128
Char embedding size	128
Tag embedding size	128
Lstm units	128
Char lstm units	128

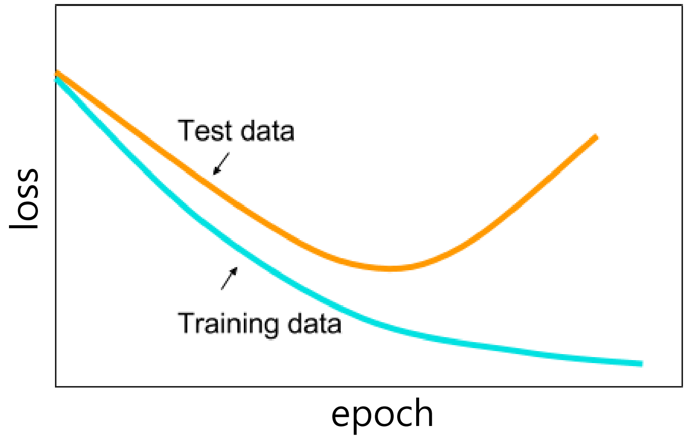
88.1035

Hyper-parameters tuning

training data score

test data score

Overfitting



Model

MODEL 1

Training data : 90000

training data score

test data score

MODEL 2

Training data : 80000

Validation data : 10000

validation data score

test data score

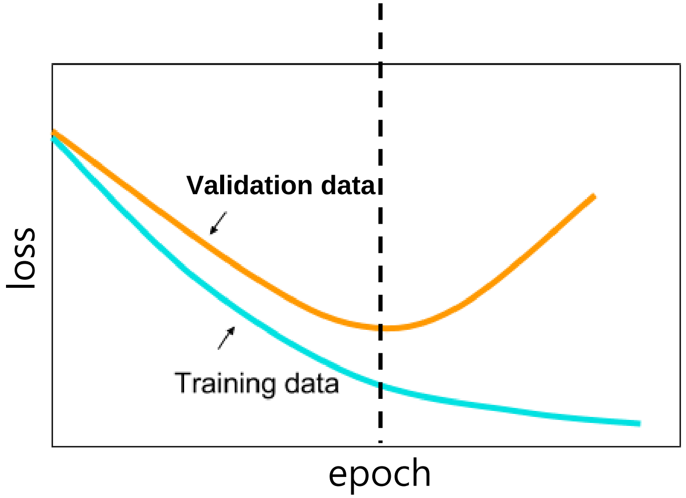
Model

MODEL 2

Training data : 80000

Validation data : 10000

10 epochs



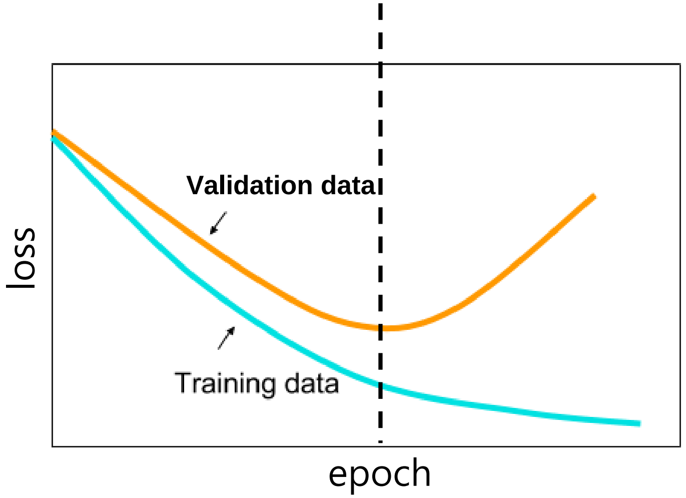
Model

MODEL 2

Training data : 80000

Validation data : 10000

10 epochs



MODEL 1

Training data : 90000

Sumbit

10 epochs ± 1 epoch

Model

Tuning Tuning Submit Tuning Submit
Submit Submit Submit
Submit Submit Tuning Submit Submit Tuning
Submit Submit Tuning Submit
Tuning Submit Submit Submit Tuning Tuning

88.1035 → 88.2567

Model

Changing optimzer

AdamOptimizer

↓

GradientDescentOptimizer

MomentumOptimizer

RMSPropOptimizer

Model

Changing optimzer

AdamOptimizer

GradientDescentOptimizer

MomentumOptimizer

RMSPropOptimizer

Tuning
Tuning Submit Tuning
Submit
Submit Submit
Submit
Submit Submit Tuning Submit
Tuning Submit
Submit Submit
Submit Tuning
Tuning Submit
Submit
Submit
Tuning Tuning

Model 1 score

88.7138
⋮
89.2098
⋮
89.8737
⋮
90.1050
⋮
90.2499

Model

MODEL 2

Training data : 80000

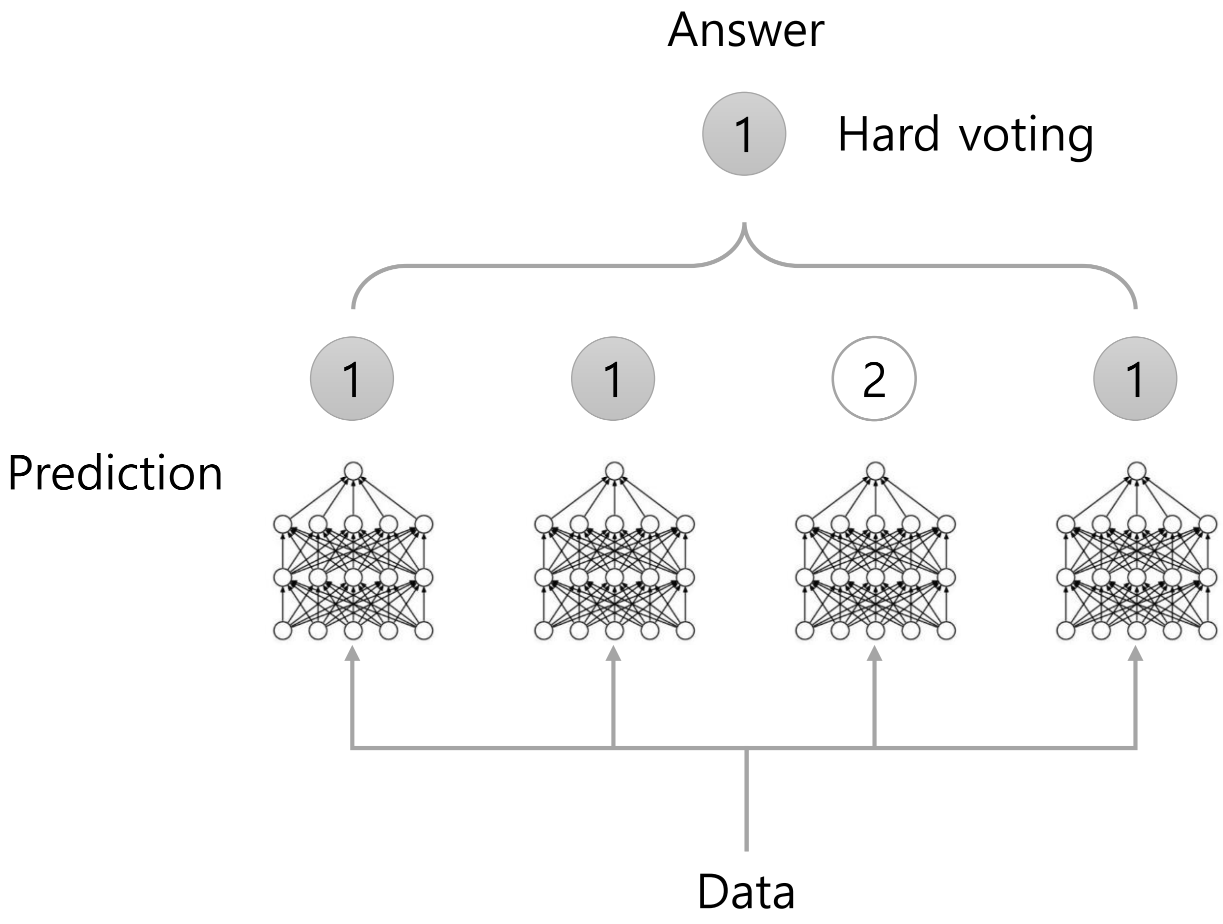
Validation data : 10000

Ensemble MODEL

Training data : 80000

Validation data : 10000

Ensemble of N different models



Model

Ensemble
MODEL

Ensemble of three different models

90.2499

↓

90.4219

Model

Ensemble MODEL

Ensemble of five different models

Ensemble MODEL

Training data : 90000

~~Validation data : 10000~~

Ensemble of N different models

THANK YOU