Natural Language Processing with Disaster Tweets

Jiaqi Liu

Jilin University

2021-07-23

Overview

Problem Definition

- Background
- Data Introduction

Data Process

- Word Spliting
- Encode the words

Model Construction

- The Embedding Layer
- TextCNN

Model Training

- Training Settings
- Training Process

Result

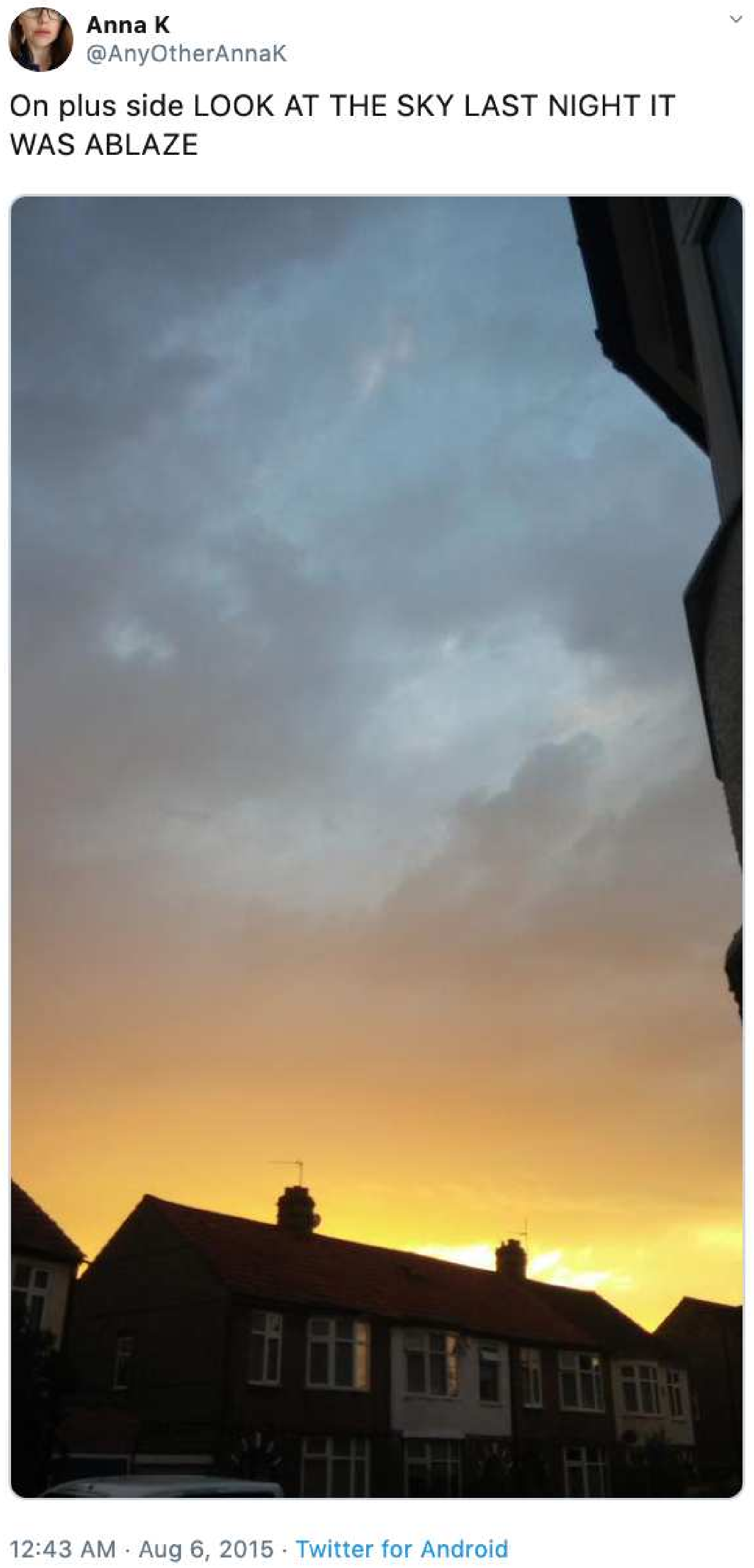
- Result

Problem Definition
Background
Data Introduction
Data Process
Model Construction
Model Training
Result

Problem Definition

Background

Twitter has become an important communication channel in times of emergency. But, it's not always clear whether a person's words are actually announcing a disaster. Take the tweeter in figure 1 as as example: Although the author used word "ablaze", clearly it wasn't about an incident. This is quite clear to human, but not to computers. Our goal is to develop a model that predicts whether a tweet is about a disaster or not.



Anna K
@AnyOtherAnnaK

On plus side LOOK AT THE SKY LAST NIGHT IT WAS ABLAZE

12:43 AM · Aug 6, 2015 · Twitter for Android

Data Introduction

file name	size	line number	columns
$train.csv$	$965KB$	7613	$id,keyword,location,text,target$
$test.csv$	$411KB$	3263	$id,keyword,location,text$

Property Explaination

- id : the identity key of every recorded tweet.
- keyword : a label representing important words in the tweet
- location : where the tweet is written
- text : the content of the tweet
- target : whether the tweet is related to a disaster

Data Process

Word Spliting

- The first task of the model is to turn text into word sequence.
  - Mainly based on regex matching.

Normal regex matching.

- List all the possible seperations and replace them with blanks.
- Split the tweet according to blanks.

Consider the vocabulary deformation.

- Match and replace them with their original forms.
  - won't → will not
  - can't → can not
  - …

Encode the words

Encode the words mentioned so that we can further turn it into vectors.

Use the pre-trained word vector&dictionary GloVe developed by Stanford.

- Include 400000 words.
- PAD & UNK also indexed.
- The proportion of recruited words is shown in Figure 1

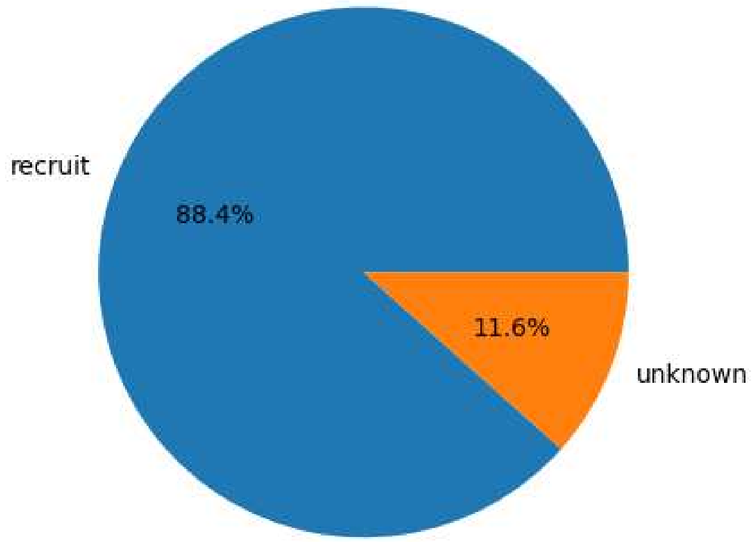


Figure 1

Model Construction

The Embedding Layer

Problem Definition
Data Process
Model Construction
The Embedding Layer
TextCNN
Model Training
Result

- Pretrained word vector GloVe.
- Turning word into vector of 100 dimensions.
- Based on co-occurence matrix.

GloVe

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

- It can be seen that the ratio represents the correlation between words.

Ratio	word j,k related	word j,k non-related
word i,k related	close to 1	very big
word i,k non-related	very small	close to 1

- As word vector also represents correlation between words, $\exists F$,so that:

$$F(w_i, w_j, w_k) = \frac{P_{ik}}{P_{jk}}$$

GloVe

$$F(w_i, w_j, w_k) = \frac{P_{ik}}{P_{jk}}$$

(consider i,j without k)

$$F(w_i - w_j, w_k) = \frac{P_{ik}}{P_{jk}}$$

(Right is a scalar)

$$F((w_i - w_j)^\top w_k) = \frac{P_{ik}}{P_{jk}}$$

$$F({w_i}^\top w_k - {w_j}^\top w_k) = \frac{P_{ik}}{P_{jk}}$$

(Consider turning minus form into fraction)

$$exp({w_i}^\top w_k - {w_j}^\top w_k) = \frac{exp({w_i}^\top w_k)}{exp({w_j}^\top w_k)} = \frac{P_{ik}}{P_{jk}}$$

$$exp({w_i}^\top w_k) = P_{ik} \quad exp({w_j}^\top w_k) = P_{jk}$$

$${w_i}^\top w_k = log(\frac{X_{ik}}{X_i}) = logX_{ik} - logX_i$$

$$({w_i}^\top w_k = {w_k}^\top w_i)$$

$$logX_{ik} = {w_i}^\top w_k + b_i + b_k$$

$$J = \sum_{ik} ({w_i}^\top w_k + b_i + b_k - logX_{ik})^2$$

$$J = \sum_{ik} f(X_{ik})({w_i}^\top w_k + b_i + b_k - logX_{ik})^2$$

TextCNN

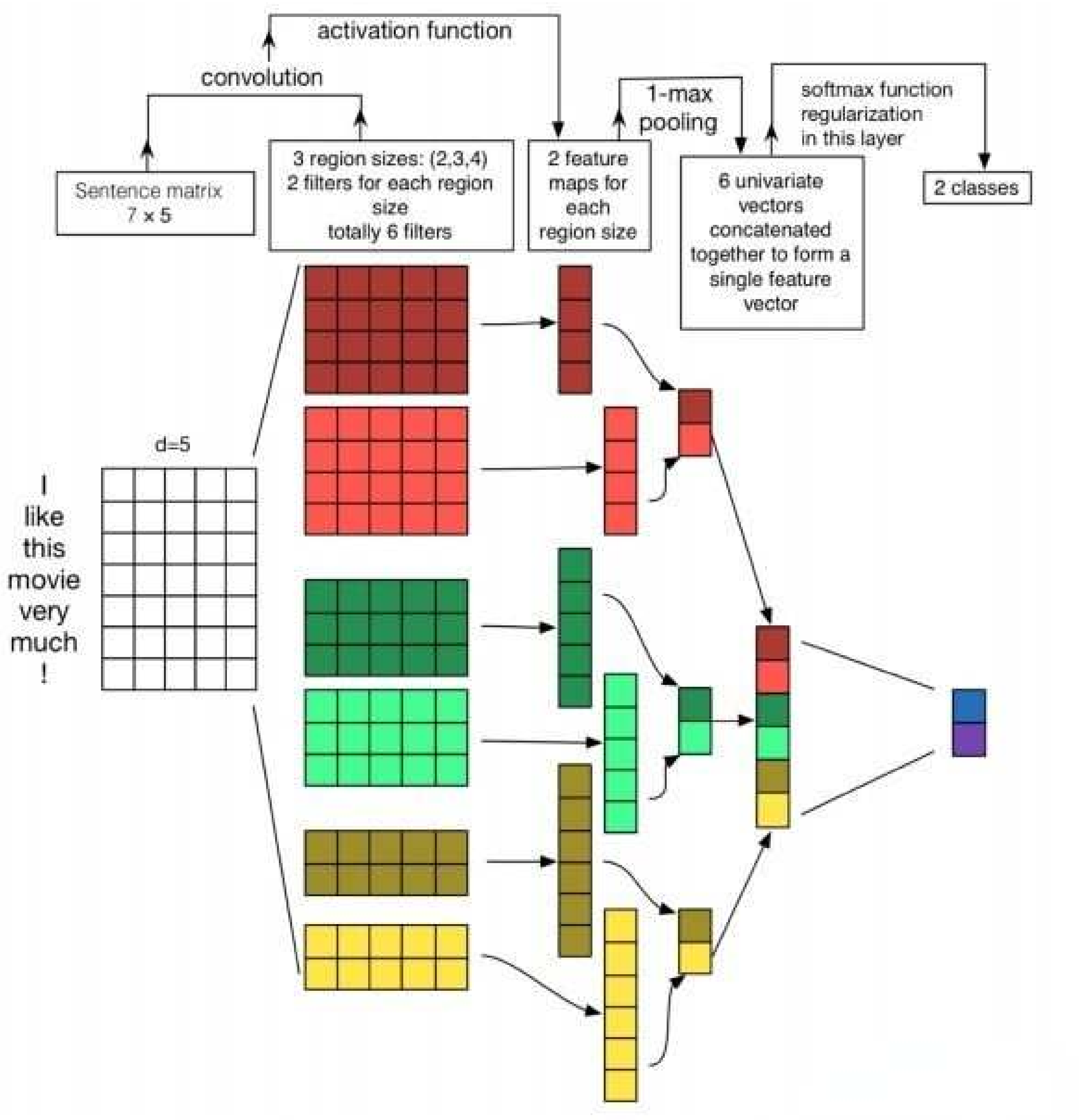

activation function
convolution
1-max pooling
softmax function regularization in this layer
Sentence matrix 7 x 5
3 region sizes: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
6 univariate vectors concatenated together to form a single feature vector
2 classes
d=5
I
like
this
movie
very
much
!

Figure 2

Model Training

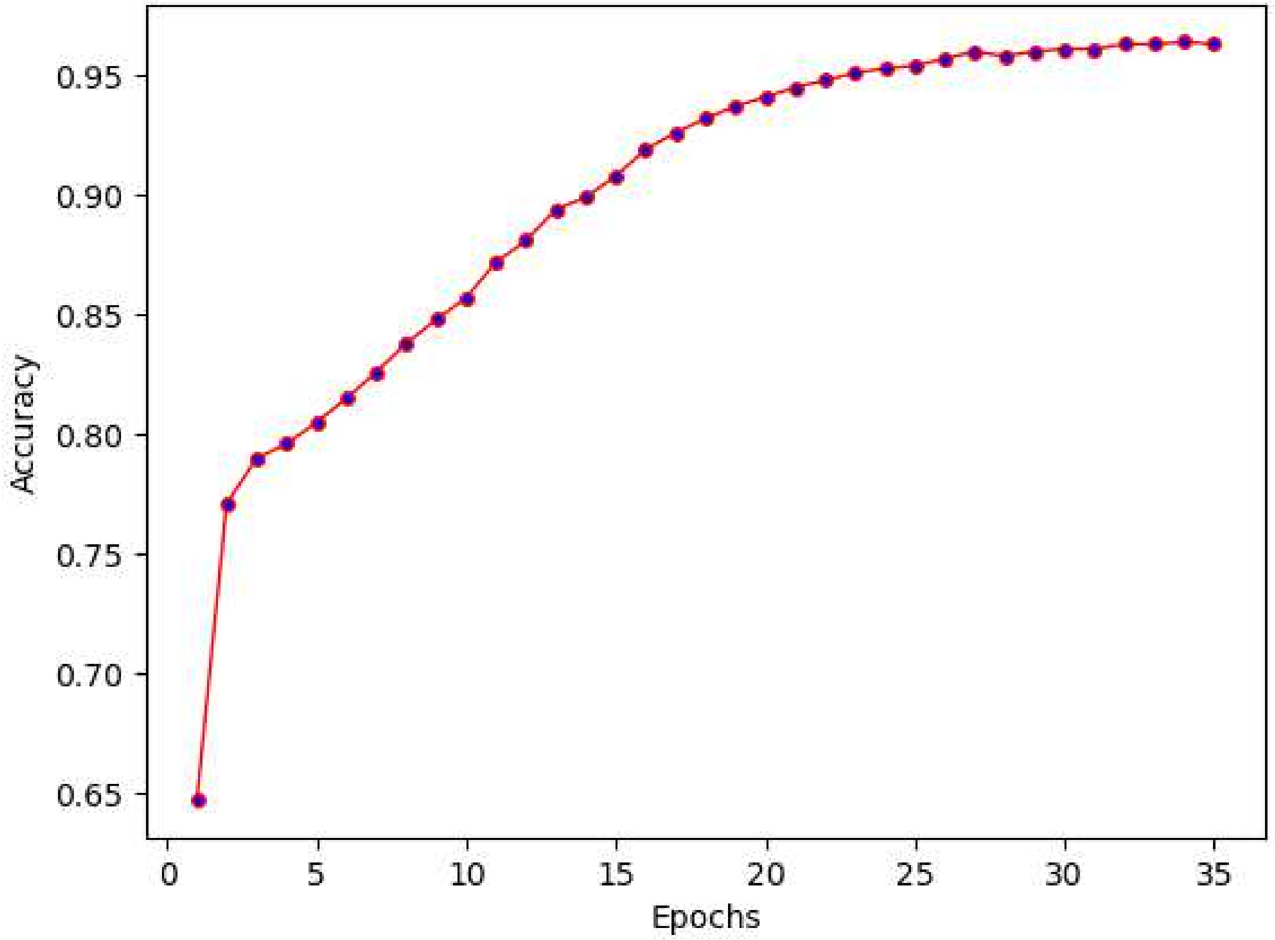
Training Settings

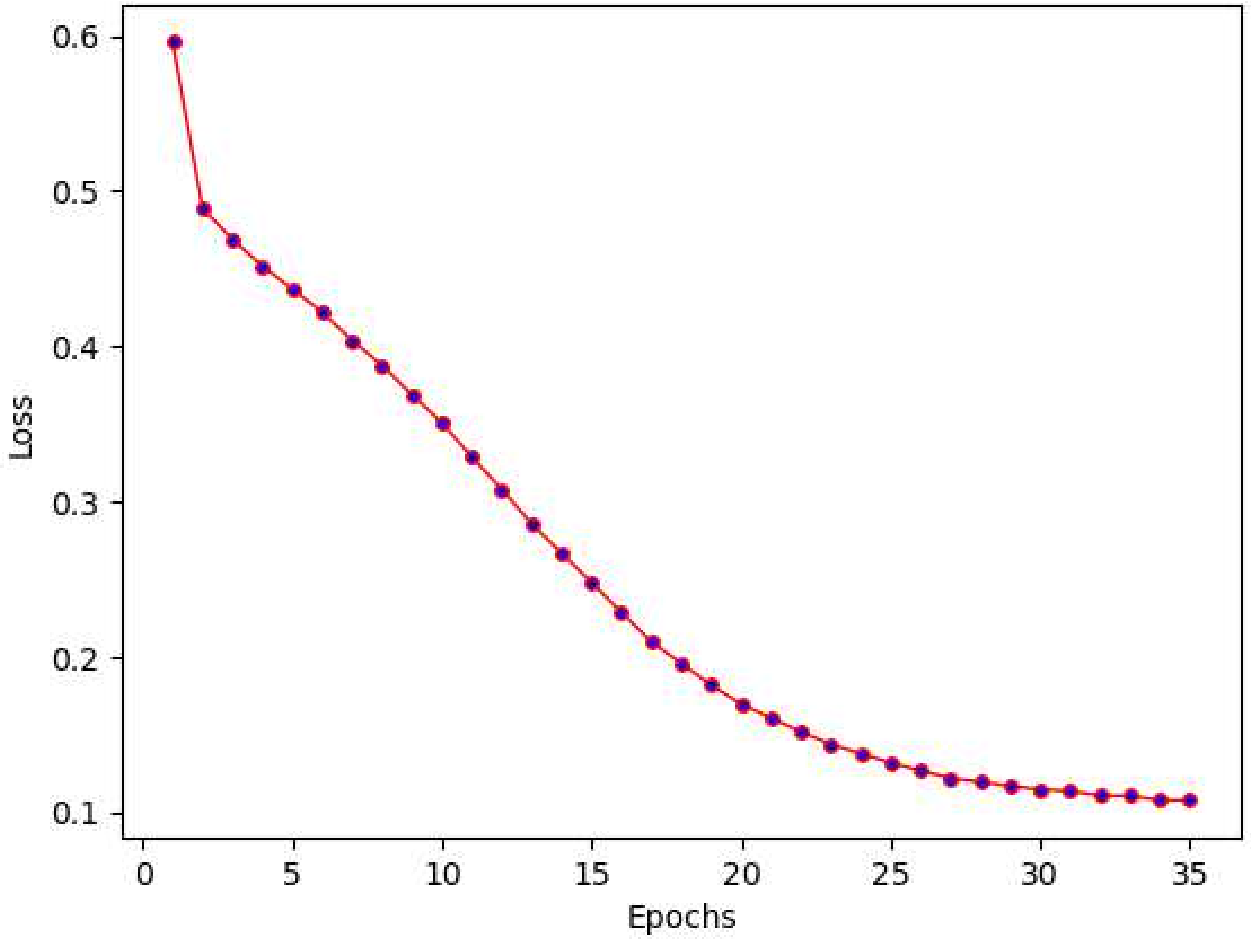
Training Settings are listed as followed:

- $batch\_size = 10$
- $learning\_rate = 0.005$
- $loss = BinaryCrossEntropy$
- $optimizer = AdamOptimizer$
- $accuracy = (\sum_{i=0}^{N} 1 - |\frac{1}{2}(sign(\hat{y}_i - 0.5) + 1) - y_i|)/N$

Training Process

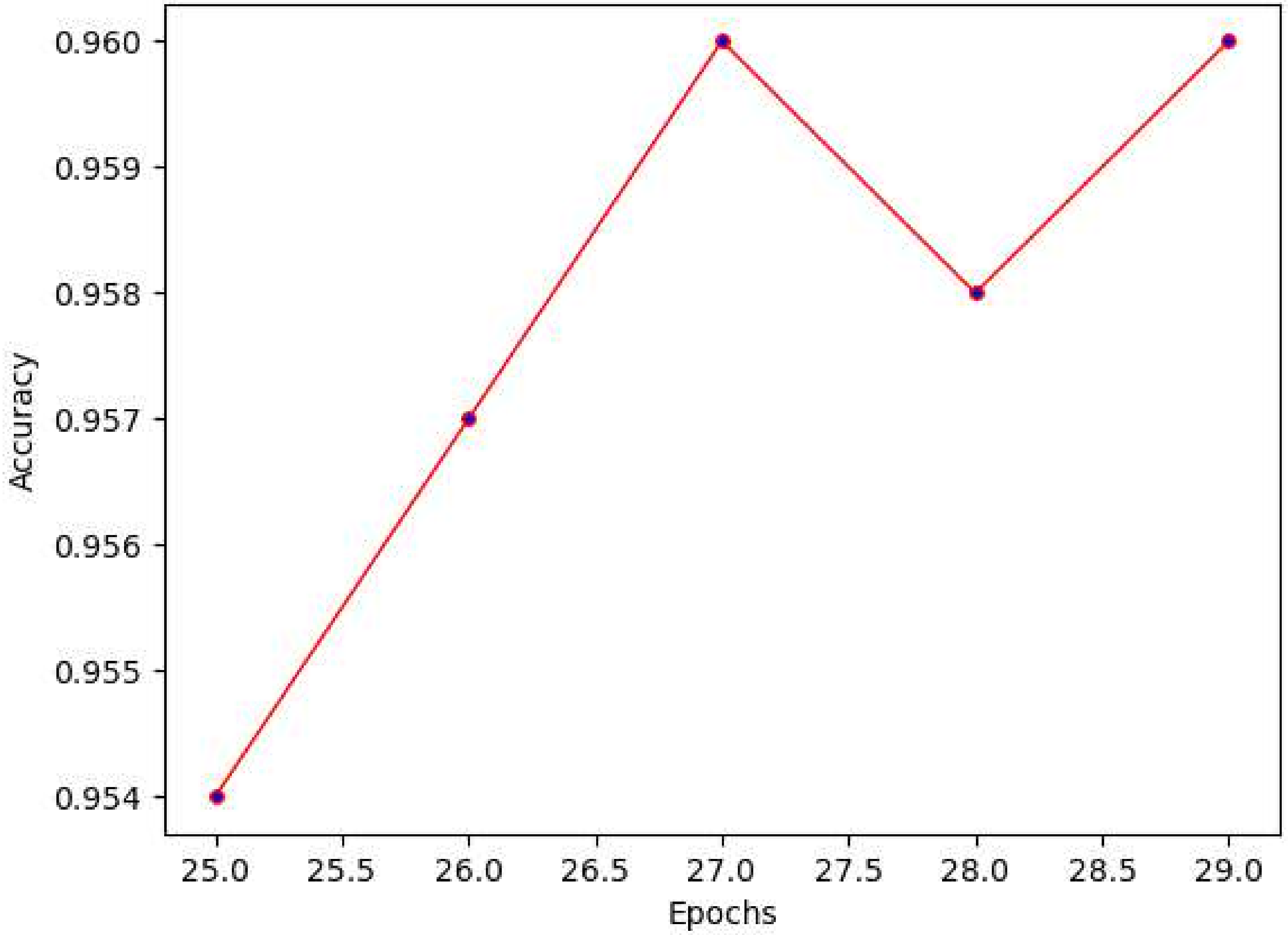
Accuracy & Loss during the training process is as the figures show:





Training Process

- An interesting phonomenonhappened near $epochs = 27$, as the figure shows.



- By testing epochs near 27, we can find the best epochs.

Training Process

Epochs	Accuracy
26	0.76340
27	0.78455
28	0.77811
29	0.76371
30	0.77106

- $epochs = 27$

Result

Result

- Final accuracyreahces 0.78455 with 27 epochs.
- Rank:2349/3625

Contact Information

Jiaqi Liu
Department of Computer Science and Technology
Jilin University, China

✉ jqliu@tulip.academy

⌂ Team for Universal Learning and Intelligent Processing