Reference https://http	/stackoverflow.com /missinglink.ai/gu	/kiVQuk1.png		J	
earning-	edding Layer works gh this blog, if	uides/keras/keras S vou have anv	s-conv1d-w	on usina p	onvolutional-neural-networks-keras/ oredefined Embedding values in
Conv1D_v			Input_Te	ext: InputLaye	n/use-word-embedding-layers-de
	with_filter_size_M: C	concatenate1	_above_tlu	•	ers: Concatenate
Conv1D	_with_filter_size_i: (	Conv1D Con	v1D_with_:	er1: MaxPooli  filter_size_j: (  ree_conv_lay	
				er2: MaxPooli  V  Silter_size_P:	
			DropC	en: Flatten	
·	mgur.com/fv1GvFJ.pn are Conv1D layers		Output	Layer: Dense	sizes, there is no restriction on this.
3. You ( 4. Don' ( Only )	concatenate layer  can use any pool s  t use more than 16  recommendation if  can use any number	size and stride for the following for the following for the following strength of layers after for the following strength of	for maxpoo Conv laye omputing p	ling layer. r becuase i <sup>s</sup> ower )	annels.  Twill increase the no of params.
## Embedd from keras embedding import nur import ter from tenso	<pre>ing layer s.layers import En layer = Embedding mpy as np nsorflow as tf orflow.keras.preps</pre>	mbedding g(len(tokenizer.	word_index mport Toke	k)+1, 50, em enizer	al/lib/python3.7/dist-packages (2.8.0) abeddings_initializer=tf.keras.initiali
<pre>from tenso from tenso from tenso from tenso from tenso input_laye embed = er #tf.keras</pre>	orflow.keras.prep: orflow.keras.mode: orflow.keras.laye: orflow.keras.call; orflow.keras.optin  er = Input(shape=  mbedding_layer(input) .layers.Conv1D(	rocessing.sequen  ls import Model  rs import Input, backs import Moden  mizers import Aden  (maxlen))  put_layer)	ce import  Dense, ConvelCheckpoi	pad_sequend v1D,Flatten, int,TensorBo	Embedding, MaxPool1D, concatenate, Dropout ard, EarlyStopping
<pre># kerne # acti # kerne # kerne # conv1D_m = Conv1D_n = Conv1D_o = second_lay</pre>	el_size = An intervation=relu, el_initializer=he el_regularizer='t.  = Conv1D(25,4,act: = Conv1D(30,4,act: = Conv1D(35,4,act: yer = concatenate	ger or tuple/lis _normal, f.keras.regulari ivation="relu",k ivation="relu",k ivation="relu",k ([Conv1D_m,Conv1	zers.12' ernel_init ernel_init	ngle integer  cializer =tf  cializer = tf	e (i.e. the number of output filters in a, specifying the length of the 1D converse.  E.keras.initializers.he_normal(), kernel lef.keras.initializers.he_normal(), kernel lef.keras.initializers.he_n
Conv1D_i = Conv1D_k = third_laye max_pool_2	<pre>= Conv1D(30,3,act: = Conv1D(35,3,act: er = concatenate( 2 = MaxPool1D(3)(f)</pre>	ivation="relu", k ivation="relu", k tivation="relu", [Conv1D_i,Conv1D] third_layer) ,activation='rel	ernel_initkernel_ini	cializer = t itializer = _k])	<pre>cf.keras.initializers.he_normal(), kerned cf.keras.initializers.he_normal(), kerned tf.keras.initializers.he_normal(), kerned cf.keras.initializers.he_normal(), kerned cf.keras.initializers.he_normal()</pre>
dense_laye		ivation="relu",k	x",kernel_	_initializer	ef.keras.initializers.he_normal())(droper
<pre>input_1 ( embedding convld (C</pre>	del"  pe)  InputLayer)  (Embedding)  onv1D)	[(None, 962 (None, 9622 (None, 9619	22)] 2, 50) 3, 25)	0 5348350 5025	Connected to  []  ['input_1[0][0]']  ['embedding[0][0]']
	<pre>(Conv1D) te (Concatenate)  ngld (MaxPooling1 (Conv1D)</pre>	(None, 9619 (None, 9619 (None, 9619  D) (None, 3206 (None, 3204 (None, 3204	9, 35) 9, 90) 5, 90)	6030 7035 0 0 6775 8130	<pre>['embedding[0][0]']  ['embedding[0][0]']  ['conv1d[0][0]',    'conv1d_1[0][0]',    'conv1d_2[0][0]']  ['concatenate[0][0]']  ['max_pooling1d[0][0]']</pre>
	te_1 (Concatenate  ngld_1 (MaxPoolin  (Conv1D)  Flatten)		58, 90) 5, 30)	9485 0 0 8130 0	<pre>['max_pooling1d[0][0]']  ['conv1d_3[0][0]',    'conv1d_4[0][0]',    'conv1d_5[0][0]']  ['concatenate_1[0][0]']  ['max_pooling1d_1[0][0]']  ['conv1d_6[0][0]']  ['flatten[0][0]']</pre>
dense (De dense_1 (	nse)  Dense)  ===================================	(None, 64) (None, 20)  ,350		2046784	['dropout[0][0]'] ['dense[0][0]']
checkpoint early_stop log_dir = tensorboa: callbacks model.comp	orflow_addons.met:  t = ModelCheckpoin  p = EarlyStopping  "logs"  rd = TensorBoard(:  =[checkpoint,ear:  pile(loss='catego:	rics import F1Sc nt(filepath='bes (monitor="val_ac log_dir=log_dir,) ly_stop,tensorbo	t_model_1.curacy",mo	ode='max',pa _freq=1,writ nizer=Adam(l	e_graph=True) earning_rate=0.001), metrics=['accurac
history =  poch 1/15 usr/local reset_sta n depreca m.reset_ poch 1: v 21/221 - ral_f1_sco poch 2/15	model.fit(x_train /lib/python3.7/di tes()` method; re ted to improve AP state() al_accuracy impro 940s - loss: 1.52 re: 0.4986 - 940s	st-packages/kera name it to `rese I consistency. ved from -inf to 23 - accuracy: 0 /epoch - 4s/step	=15, verbos  as/engine/s et_state()  0.49862, 0.5410 - file	training.py (without to saving mode) 1_score: 0.5	<pre>ion_data=(x_test,y_test),batch_size =6  22034: UserWarning: Metric F1Score impl the final "s"). The name `reset_states( el to best_model_1.h5 6410 - val_loss: 1.6694 - val_accuracy:</pre>
Spoch 2: v 221/221 - ral_f1_sco Spoch 3/15 Spoch 3: v 221/221 - ral_f1_sco Spoch 4/15 Spoch 4: v 221/221 - ral_f1_sco	al_accuracy impro 914s - loss: 1.48 re: 0.5041 - 914s al_accuracy impro 906s - loss: 1.45 re: 0.5320 - 906s al_accuracy impro 908s - loss: 1.42 re: 0.5543 - 908s	75 - accuracy: 0 /epoch - 4s/step ved from 0.50414 62 - accuracy: 0 /epoch - 4s/step ved from 0.53197 38 - accuracy: 0	1 to 0.531 1 to 0.531 1 to 0.531 2 to 0.5542 3 to 0.5542	1_score: 0.9 97, saving r 1_score: 0.9	model to best_model_1.h5 5560 - val_loss: 1.6810 - val_accuracy: model to best_model_1.h5 5721 - val_loss: 1.5736 - val_accuracy: model to best_model_1.h5 5851 - val_loss: 1.5562 - val_accuracy:
ral_f1_sco poch 5/15 poch 5: v 21/221 - ral_f1_sco poch 6/15 poch 6: v 21/221 - ral_f1_sco poch 7/15	re: 0.5543 - 908s  al_accuracy impro 902s - loss: 1.39 re: 0.5551 - 902s  al_accuracy impro 907s - loss: 1.36 re: 0.5798 - 907s  al_accuracy did n	/epoch - 4s/step  ved from 0.55428 25 - accuracy: 0 /epoch - 4s/step  ved from 0.55513 68 - accuracy: 0 /epoch - 4s/step  ot improve from	3 to 0.555; 0.6023 - f; 3 to 0.579; 0.6126 - f;	13, saving r 1_score: 0.6 77, saving r 1_score: 0.6	model to best_model_1.h5 5023 - val_loss: 1.5702 - val_accuracy: model to best_model_1.h5 5126 - val_loss: 1.5333 - val_accuracy:
221/221 - ral_f1_sco poch 8/15  Spoch 8: v 221/221 - ral_f1_sco accuracy_a f1Score_a:	897s - loss: 1.35 re: 0.5528 - 897s al accuracy did n	19 - accuracy: 0 /epoch - 4s/step  ot improve from 24 - accuracy: 0 /epoch - 4s/step  istory['accuracy story['fl_score'	0.57977 0.6361 - f:	_	5214 - val_loss: 1.5771 - val_accuracy: 5361 - val_loss: 1.5795 - val_accuracy:
print (f18d print (loss 0.5410381 2165222, 0.5410381 22165222, 1.5222605 1938, 1.35	555557251, 0.5559 0.621414899826049 555557251, 0.5559 0.621414899826049	8, 0.63607394695 804439544678, 0. 8, 0.63607394695 863624572754, 1. .322362422943115	528198] 572055816 528198] 4561767578	6503906, 0.5	5850860476493835, 0.6022944450378418, 0 5850860476493835, 0.6022944450378418, 0 7747192382812, 1.392497181892395, 1.366
plt.figure plt.plot(: plt.scatte plt.title plt.xlabe	e(figsize = (10,7) range(len(accuracy er(range(len(accu: ('Accuracy plot') l('Epochs') l('Accuracy') d()	)) y_array)),accura			
0.64	Accuracy Accuracy Points	.show>	y plot		
0.58					
plt.plot(: plt.scatte plt.title plt.xlabe:	e(figsize = (10,7); range(len(f1Score)er(range(len(f1Score))))))))))))))))))))))))))))))))))))	_array)),f1Score	_array,lak		
plt.legend plt.grid() plt.show function	d()	.show>	plot		
0.60					
0.56	-	2 3 Epoch	4 hs	5	6 7
<pre>import mat epochs =1!  plt.figure plt.plot(: plt.scatte plt.title</pre>	e(figsize = (10,7) range(len(loss_ar: er(range(len(loss) ('loss') l('Epochs') l('loss') d()	s plt  ))  ray)),loss_array			
	matplotlib.pyplot	.show>	is		loss
1.450					
	: Using 1D conv	yolutions with		er embed	ding
	Jse 1D-con	IVOIUTIONS et sliced by characters	:!		
Input change = alphabet s = 70 channe	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 1 250 0 0 0 0 0 1 0 0 0 1	0 1 Input c	1 image hannel: 70 channel: 1 shape: (1, 1, 136)
<b>♦₽</b> ■◆	e the some papers ng Zhang, Junbo Zh ication.NIPS 2015 n Kim, Yacine Jerr 16	nao, Yann LeCun. nite, David Sonta Kolter, Vladlen equence Modeling nar embeddings ht	Character ag, Alexan Koltun. A	der M. Rush	olutional Networks for Text  Character-Aware Neural Language Models  Evaluation of Generic Convolutional and
Here are 1. Xiai Classif 2. Yoo AAAI 20: 3. Shac Recurrer 4. Use	sent_input	: InputLayer  t_char_embedd:	Embeddir	ıg	
Here are 1. Xiai Classif 2. Yoo AAAI 20: 3. Shac Recurrer 4. Use embeddi	ng_Layers_to_get	er size n: Convi	1D		
Here are 1. Xiai Classif 2. Yoo AAAI 20: 3. Shac Recurrer 4. Use embeddin	ng_Layers_to_get Conv1D_with_filt	<b>\</b>	1D		
Here are 1. Xiai Classif 2. Yoo AAAI 20: 3. Shac Recurrer 4. Use embeddin	Conv1D_with_filt	er_size_m: Conv			
Here are 1. Xiai Classif 2. Yoo AAAI 20: 3. Shac Recurrer 4. Use embeddin	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt	er_size_m: Conv			
Here are 1. Xiai Classif 2. Yoo AAAI 20: 3. Shac Recurrer 4. Use embeddin	Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten	er_size_m: Conv			
Here are 1. Xiai Classif 2. Yoo AAAI 20: 3. Shac Recurrer 4. Use embeddin	Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut	er_size_m: Conv			
Here are 1. Xiai Classif 2. Yoo AAAI 20: 3. Shac Recurrer 4. Use embeddi	Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  DropOut  Densel  OutputLa  _test,y_train,y_te	er_size_m: Conv  I MaxPooling1D  er_size_k: Convi  er_size_t: Convi  i: MaxPooling1D  :: Flatten  :: Dropout  :: Dense  est = train_test  filters='!"#\$%&(	ID		
Here are 1. Xial Classif 2. Yook AAAI 20: 3. Shack Recurrer 4. Use embedding  ## Tokeni x_train = x test = t #Applying length_of length_of	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  test,y_train,y_test  char = Tokenizer(star.fit_on_texts)  ze them tokenize_char.text	er_size_m: Conv  :: MaxPooling1D  er_size_k: Conv1  :: er_size_t: Conv1  :: Dropout  :: Dropout  :: Dense  :: Dense  est = train_test  :: crain_test  :: xts_to_sequences  :xts_to_sequences	ID  (x_train) x_test)		
Here are 1. Xiai Classif; 2. Yoo AAAI 20: 3. Shac Recurrer 4. Use embedding  ## Tokeni: x_train = x test = i #Applying length_of length_	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa test, y_train, y_test tokenize_char.text  ze them tokenize_char.text tokenize_char.text  ze them tokenize_char.text  and ingsentences = [lensentences.sort()sentences = np.as  _layer_char = Ember er = Input(shape= mbedding_layer_char  1D(64,3,activation)  1D(64,3,activation)	er_size_m: Conv  :: MaxPooling1D  er_size_k: Conv1  :: er_size_t: Conv1  :: Dropout  :: Dropout  :: Dense  :: Dense  est = train_test  (x_train)  exts_to_sequences (x_train)	ID  (x_train) x_test)  rain] entences)  ize_char.v	<pre>vord_index) + er =tf.keras</pre>	{ }~\t\n',char_level= True,oov_token='\displaystyle token='\displaystyle
Here are  1. Xial Classif 2. Yood AAAI 20 3. Share Recurred 4. Use embedding  ## Tokeni x_train = x_test = fe  #Applying length_of lengt	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa test,y_train,y_te char = Tokenizer(: char.fit_on_texts  ze them tokenize_char.text  ze them tokenize_char.text  and the missentences = [len tokenize_char.text  char = Tokenizer(: char.fit_on_texts  ze them tokenize_char.text  and tokenize_char.text  char = Tokenizer(: char.fit_on_texts  ze them tokenize_char.text  and tokenize_char.text  char.fit_on_texts  char.fit_on_texts  ze them tokenize_char.text  char.fit_on_texts  ze them tokenize_char.text  char.fit_on_texts  ze them tokenize_char.text  and tokenize_char.text  char.fit_on_texts  ze them tokenize_char.texts  ze them tokenize_char.texts  ze them tokenize_char.texts  ze them	er_size_m: Conv  :: MaxPooling1D  :: er_size_k: ConvI  :: er_size_t: ConvI  :: Dropout  :: Dropout  :: Dense  :: Dense  est = train_test  :: train_test  :: train_test  :: train_test  :: propout  ::	ID  (x_train) x_test)  rain] entences)  ize_char.v  initialize initialize	<pre>vord_index) + er = tf.keras er = tf.keras er = tf.keras</pre>	{ }~\t\n',char_level= True,oov_token='\displaystyle
Here are 1. Xial Classif; 2. Yook AAAI 20: 3. Share Recurrer 4. Use embedding  Embedding length_of;	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  test,y_train,y_te  char = Tokenizer(: char.fit_on_texts  ze them tokenize_char.text  ze them_sentences = [len _sentences.sort() _sentences = np.a:  layer_char = Ember er = Input(shape= mbedding_layer_char  1D(64,3,activation)  1D(64,3,activation)  1D(64,3,activation)  1D(64,3,activation)  1D(64,3,activation)  1D(64,3,activation)  2 = MaxPool1D(5) (: 1D(64,3,activation)  3 = MaxPool1D(5) (: 1D(64,3,activation)  4 = Dense(256,activation)  5 = Dense(256,activation)  6 = Dense(256,activation)  6 = Dense(256,activation)  6 = Dense(256,activation)	er_size_m: Converged and the c	ID  (x_train) x_test)  rain] entences)  ize_char.v  initialize initialize initialize initialize	<pre>vord_index) + er = tf.keras er = tf.keras er = tf.keras er = tf.keras er = tf.keras</pre>	{ }~\t\n',char_level= True,oov_token='\displaystyle
Here are  1. Xial Classiff 2. Yook AAI 20 3. Shack Recurrer 4. Use embedding first_laye embedding first_laye embed = er m1 = Convi max_pool_ i1 = Convi	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  test, y_train, y_test  char = Tokenizer (sechar.fit_on_texts)  ze them tokenize_char.text  padding sentences = [len sentences = rp.as  layer_char = Ember  er = Input (shape= embedding_layer_char  layer_char = Ember  in [lo (64, 3, activation of the content of the conten	er_size_m: Conv  :: MaxPooling1D  er_size_k: ConvI  :: MaxPooling1D  :: Flatten  :: Dropout  :: Dropout  :: Dense  est = train_test  :: Latter  :: Dropout  :: pense  est = train_test  :: pense  est	ID  Split(x,)  (x_train) x_test)  rain] entences)  ize_char.v  initialize initialize initialize initialize  tput_layer  tput_layer	<pre>vord_index)+  vord_index)+  er = tf.keras  er</pre>	{  }~\t\n',char_level= True,oov_token='! -1,41, embeddings_initializer=tf.keras.: -1,41, embeddings_initializer=tf.kera
x_train, x_ tokenize_o AAAI 20: 3. Shace Recurred 4. Use embedding length_of	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  tokenize_char.text  char = Tokenizer(: char.fit_on_texts  ze them tokenize_char.text  padding sentences = [len sentences = np. a: layer_char = Embo er = Input (shape= mbedding_layer_char 1D (64, 3, activation) 2 = MaxPool1D (5) (: 1D (64, 3, activation) 3 = MaxPool1D (5) (: 1D (64, 3, activation) 4 = Dense (256, activation) 5 = Dense (20, activation) 6 = Dense (20, activation) 7 = Dense (20, activation) 7 = Dense (20, activation) 8 = Dense (20, activation) 9 = Dense (20, activation) 1D (Conv1D)	er_size_m: Conv  cr_size_m: Conv  cr_size_k: Conv  er_size_k: Conv  cr_size_t: Conv  cr_siz	TD  TD  TD  TD  Tain  (x_train)  x_test)  rain  entences)  ize_char.v  initialize	<pre>c=&gt;?@[\\]^_`  vord_index)+  er =tf.keras  er = tf.keras  er = tf.keras  er = tf.keras  f. ler = tf.keras  at = tf.keras</pre>	{  }~\t\n',char_level= True,oov_token='! -1,41, embeddings_initializer=tf.keras1,41, embeddings_initializer=t
## Tokeni x_train, x tokenize_d 4. Use embeddin  Embeddin  ## Tokeni x_train = 1  #Applying length_of leng	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa test,y_train,y_te char = Tokenizer(: char.fit_on_texts  ze them	er_size_m: Conv er_size_m: Conv  : MaxPooling1D  er_size_k: Conv  : er_size_t: Conv  : Tatten  : Dense  ::	ID  Split(x, y)  (x_train) x_test)  rain] entences)  ize_char.v  initialize	<pre>Param # ====================================</pre>	{  }~\t\n',char_level= True,oov_token='\ -1,41, embeddings_initializer=tf.keras1,41, embeddings_initializer=tf.keras1,41
*# Tokenize of tok	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  MaxPoolLayer2  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa test,y_train,y_tc  char = Tokenizer(: char.fit_on_texts  ze them	er_size_m: Conv  i: MaxPooling1D  er_size_k: ConvI  er_size_t: Con	ID  ID  ID  (x_train) (x_train) (x_test)  rain] entences) ize_char.v  initialize	<pre>vord_index) + er = tf.keras er = tf.ker</pre>	<pre>### True, cov_token=' ### True, cov_tok</pre>
## Tokenix x_train, x tokenize_d  ## Tokeni x_train = x x_train, x  tokenize_d  ## Tokeni x_train = x	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  test,y_train,y_te char = Tokenizer(s char.fit_on_texts  ze them tokenize_char.text  chenize_char.text  ze them tokenize_char.text  char.fit_on_texts  ze them tokenize_char.text  layer_char = Embrer er = Input(shape= mbedding_layer_char  layer_char = Embrer er = Input(shape= mbedding_layer_char  lo(64,3,activation)  1 = MaxPoollD(5) (s)  10 (64,3,activation)  10 (64,3,activation)  10 (64,3,activation)  11 = Dense (256,activation)  12 = MaxPoollD(5) (s)  13 = MaxPoollD(5) (s)  14 = Dense (256,activation)  15 = Convert  16 = Dense (20,activation)  17 = Dense (20,activation)  18 = Convert  19 = Dense (20,activation)  19 = Dense (20,activation)  10 = Dense (20,activation)  11 = Dense (256,activation)  12 = Dense (256,activation)  13 = MaxPoollD(5) (s)  14 = Dense (256,activation)  15 = Dense (256,activation)  16 = Dense (256,activation)  17 = Dense (256,activation)  18 = Dense (256,activation)  19 = Dense (256,activation)  10 = Dense (256,activation)  11 = Dense (256,activation)  12 = Dense (256,activation)  13 = MaxPoollD(5) (s)  14 = Dense (256,activation)  15 = Dense (256,activation)  16 = Dense (256,activation)  17 = Dense (256,activation)  18 = Dense (256,activation)  19 = Dense (256,activation)  20 = MaxPoollD(5) (s)  21 = Dense (256,activation)  22 = MaxPoollD(5) (s)  23 = MaxPoollD(5) (s)  24 = Dense (256,activation)  25 = Dense (256,activation)  26 = Dense (256,activation)  27 = Dense (256,activation)  28 = Dense (256,activation)  29 = Dense (256,activation)  20 = Dense (256,activation)  21 = Dense (256,activation)  22 = Dense (256,activation)  23 = Dense (256,activation)  24 = Dense (256,activation)  25 = Dense (256,activation)  26 = Dense (256,activation)  27 = Dense (256,activation)  28 = Dense (256,activation)  29 = Dens	er_size_m: Converger_size_k: Converger_size_k: Converger_size_k: Converger_size_k: Converger_size_t: C	ID  ID  ID  ID  IS  Selection  Selection  (x_train)  x_test)  rain]  entences)  ize_char.v  initialize  initialize	<pre>Param #  Param #  initializer =  initializer =  initializer =  initializer =  1681  7936  12352  0  12352  0  12352  0  12352  0  12352  0  1278208  5140 </pre>	<pre>## Comparison of the control of</pre>
Here are  1	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  MaxPoolLayer2  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa test,y_train,y_tr	er_size_m: Converger_size_k: Converger_size_k: Converger_size_k: Converger_size_t: C	ID  ID  ID  ID  Split(x,y)  _split(x,y)  _split(x,y)  (x_train) x+,/:;<  (x_train) x_test)  rain] entences)  ize_char.v  initialize initializ	Param #  Par	<pre>### Comparison of the final "s" ### Comparison of the final "s" #### Comparison of the final "s" #### Comparison of the final "s" #### Comparison of the final "s" ##### Comparison of the final "s" ##### Comparison of the final "s" ###################################</pre>
Here ari 1. Xia Classif 2. Yoo AAAI 20: 3. Sha Recurret 4. Use embeddin  Embeddin  Embeddin   X train, X  tokenize tokenize  ** Train, X  tokenize tokenize  ** Train  ** Train  #Applying length_of length_of length_of length_of length_of length_of length_of length_of if at Convi  max_pool  i1 = Convi  max_pool  i1 = Convi  max_pool  i1 = Convi  max_pool  i1 = Convi  convid_0  model = Mod  mode	ConvID_with_filt  ConvID_with_filt  MaxPoolLayer1  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  MaxPoolLayer2  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  test,y_train,y_tc  char = Tokenizer(: char.fit_on_texts  ze them tokenize_char.text  charifit_on_texts  ze them tokenize_char.text  padding sentences = np. a: layer_char = Emble er = Input(shape= mbedding_layer_char)  lo(64,3,activation)  10(64,3,activation)  11(64,3,activation)  12 = MaxPoollD(5) (: 11(64,3,activation)  12 = MaxPoollD(5) (: 11(64,3,activation)  13 = MaxPoollD(5) (: 11(64,3,activation)  14 = MaxPoollD(5) (: 15 = MaxPoollD(5) (: 16 = MaxPoollD(5) (: 17 = Dense(256,act)  del (inputs=first_index)  mary()  del _ 1"  pe)  =================================	er_size_m: Converger_size_m: Converger_size_k: Converger_size_k: Converger_size_t: C	ID  ID  ID  ID  IS  Split(x,)  (x,)	respectively.  The series of t	### True, oov_token='
Here are  11. Xia  Classif  2. Yoo  AAAI 20:  3. Shae  Recure  4. Use  embedding  formation  x_train,x  tokenize  tokenize  ## Tokeni  x_train = x  x_trest = c  #Applying  length_of convid  max_pool_ if = Convi  dense_laya  output_lay  model = Mod  model. summ  fodel: "mo  Layer (ty  ===================================	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  test,y_train,y_to  char = Tokenizer(: char.fit_on_texts  ze them tokenize_char.text  chenize_char.text  chenize_char.text  chenize_char.text  chenize_char.text  chenize_char.text  padding sentences = np.a:  layer_char = Emberer = Input (shape= mbedding_layer_char  layer_char = Emberer = Input (shape= mbedding_layer_char  chenize_char.text  chenize_char.text  chenize_char.text  padding sentences = np.a:  layer_char = Emberer = Input (shape= mbedding_layer_char  control = MaxPoolID(5) (input (shape= mbedding_layer_char  control = Input (shape= m	regize_m: Converting  regize_k: Converting  regize_k: Converting  regize_k: Converting  regize_k: Converting  regize_t: Converting	ID	### #### #############################	[1]-\t\n', char_level= True, cov_token='  [1,4], embeddings_initializer*tf.keras.  [2,4], kerne  [2,4], kerne  [3,4], kerne  [4,4], kerne  [5,4], kerne  [5,4], kerne  [5,4], kerne  [6,4], kerne  [7,4], ker
Here ariant in a convious and a conv	ConvID_with_filt  ConvID_with_filt  MaxPoolLayer1  MaxPoolLayer1  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  test,y_train,y_tr	er_size_m: Conv  i: MaxPoolingID  er_size_k: Convi  er_size_k: Con	ID  ID  ID  ID  ID  IS  Seplit (x, y)  (x train)  x test)  rain]  entences)  ize char.v  initialize  i	######################################	(  )-\t\n', char_level= True, cov_token='  1,41, embeddings_initializer=tf.keras.  .initializers.he_normal(seed=42), kerne .initializers.he_normal(seed=42), kerne .s.initializers.he_normal(seed=42), kern  s.initializers.he_normal(seed=42), kern  tienc=2)  cti.keras.initializers.he_normal(seed=42), kern  tf.keras.initializers.he_normal(seed=42), kern  tf.keras.initializers.glorot_normal(seed=42), kern  tto_data=(x_test,y_test), batch_size = 42634: UserWarning: Ketric FlScore implies final "s"). The name 'reset_states(seed=42), kerne  100
Here ar  1. Xiai Classif 2. Yook AAJ 2ha Recurred 4. Use embeddin  Finbeddin	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOul  Densel  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOul  Conv1D_with_filt  Flatten  DropOul  Densel  Conv1D_with_filt  Flatten  DropOul  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_conv1  Conv1D_den  Index - Token   Conv1  Conv1D_den  Conv1D_den	er_size_m: Conv  i: MaxPoolingID  er_size_k: ConvI  er_size_t: ConvI  er_size_t: ConvI  iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii	ID	######################################	(  -\t\n',char_level= True,oow_token='
Here ar  1. Xial  Classif  2. Yool  AAI 26a  Recurse  4. Use embeddin  Embeddin  Finbeddin  Convid  Co	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOul  Densel  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOul  Conv1D_with_filt  Flatten  DropOul  Densel  Conv1D_with_filt  Flatten  DropOul  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_conv1  Conv1D_den  Index - Token   Conv1  Conv1D_den  Conv1D_den	r. size_m: Conv  : MaxPooling1D  er_size_k: Conv!  er_size_t: Conv!  : MaxPooling1D  : Flatten  : Flatten  : Flatten  : Dropout  :: MaxPooling1D  : Flatten  : Flatten  : Topopout  : Dense  : State = train_test  (x_train)  : State = train_test  filters='!"#\$%%( (x_train)  : tray(length_of_s  edding(len(token (maxlen))  ar(first_layer)  n="relu", kernel_ n="relu", kernel_ n="relu", kernel_ n="relu", kernel_ n="relu", kernel_ ii)  n="relu", kernel_ iii)  n="relu", kernel_ iii)  n="relu", kernel_ iii)  n="relu", kernel_ n="relu", kernel_ n="relu", kernel_ iii)  n="relu", kernel_ iii)  n="relu", kernel_ iii)  n="relu", kernel_ n="relu", kernel_ iii)  n="relu", kernel_ iii)  n="relu", kernel_	ID	######################################	(  -\t\n',char_level= True,oov_token='
Here ar  1. Xia  Classif  2. Yoo  AAI 26a  Recent  4. Use  embeddin  Embeddin  Embeddin   Embeddin	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer1  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer2  Flatten  DropOul  Densel  Densel  conv1D_with_filt  DropOul  DropOul  Densel  CoutputLa  conv1D_with_filt  DropOul  Densel  CoutputLa  conv1D_with_filt  Flatten  DropOul  Densel  CoutputLa  conv1D_with_filt  DropOul  Densel  Densel  CoutputLa  conv1D_conv1  Conv1D_c	r. size_m: Conv  : MaxPooling1D  er_size_k: Conv!  er_size_t: Conv!  : MaxPooling1D  : Flatten  : Flatten  : Flatten  : Dropout  :: MaxPooling1D  : Flatten  : Flatten  : Topopout  : Dense  : State = train_test  (x_train)  : State = train_test  filters='!"#\$%%( (x_train)  : tray(length_of_s  edding(len(token (maxlen))  ar(first_layer)  n="relu", kernel_ n="relu", kernel_ n="relu", kernel_ n="relu", kernel_ n="relu", kernel_ ii)  n="relu", kernel_ iii)  n="relu", kernel_ iii)  n="relu", kernel_ iii)  n="relu", kernel_ n="relu", kernel_ n="relu", kernel_ iii)  n="relu", kernel_ iii)  n="relu", kernel_ iii)  n="relu", kernel_ n="relu", kernel_ iii)  n="relu", kernel_ iii)  n="relu", kernel_	ID	######################################	(!) "Ntin', char_level= True, oov_token="  1, 41, embeddings_initializer=tf.keras.  1, 41, embeddings_initializer=tf.keras.  1, 41, embeddings_initializer=tf.keras.  1, 41, embeddings_initializer=tf.keras.  1, initializers.he_normal(seed=42), kern  1, initializers.he_normal(seed=42
Here are  1. Xia  Classif  2. Yoo  A3. Sha  Recurre  4. Use embeddin  Finbeddin  Finbeddin	ConvID_with_filt  ConvID_with_filt  MaxPoolLayer1  ConvID_with_filt  ConvID_convID	er_size_m: Conv  it MaxPoolingID  er_size_k: Convi  er_size_k: Convi  er_size_t: er_size_t  er_size_t: er_si	ID	### ### #### #### ####################	(  -\c\n',char_icvel= True,cov_tokene'
Here are  1. Xia: Classif: 2. You AAI 20: AAI 20: ARE current 4. Use embedding  Finbedding  Finbedding  ** ** ** ** ** ** ** ** ** ** ** ** *	Conv1D_with_filt  Conv1D_with_filt  MaxPoolLayer  Conv1D_with_filt  Flatten  Flatten  DropOut  Densel  Conv1D_with_filt	## Size m: Conv  ## Size m: Conv  ## Size m: Conv  ## Size k: Conv  ## Size m: Conv  ## Siz	ID	### ### #### #### ####################	( -\t\n',char_lavel= True,cov_tokene')  1,41, embeddings_initializerscf.keras:  .initializers.he_normal(seed=42), kerne .initializers.he_normal(seed=42), kerne s.initializers.he_normal(seed=42), kerne s.initializers.he_normal(seed=42), kerne s.initializers.he_normal(seed=42), kerne e.initializers.he_normal(seed=42), kerne  *.initializers.he_normal(seed=42), kerne s.initializers.he_normal(seed=42), kerne  *.initializers.he_normal(seed=42), kerne
Here are  1. Xiai Classifi 2. Your ANA 120 3. Share Recurse 4. Use embedding	ConvID_with_filt  ConvID_with_filt  MaxPoolLayers  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  MaxPoolLayers  Flatten  MaxPoolLayers  Flatten  DropOut  Densel  OutputLa  Est,y_train,y_trai	## Size m: Conv  ## Size m: Conv  ## Size m: Conv  ## Size k: Conv  ## Size m: Conv  ## Siz	ID	### ### #### #### ####################	( -\t\n',char_lavel= True,cov_tokene')  1,41, embeddings_initializerscf.keras:  .initializers.he_normal(seed=42), kerne .initializers.he_normal(seed=42), kerne s.initializers.he_normal(seed=42), kerne s.initializers.he_normal(seed=42), kerne s.initializers.he_normal(seed=42), kerne e.initializers.he_normal(seed=42), kerne  *.initializers.he_normal(seed=42), kerne s.initializers.he_normal(seed=42), kerne  *.initializers.he_normal(seed=42), kerne
Here are  1. Xian  Classifor  AAI 20  3. Share  Require  4. Use  embedding  formbedding  formbedding  formbedding  firstain =  x_test = from  ## Tokenix  x_train,x  tokenize_  ## Tokenix  ## Tokenix  x_train,x  tokenize_  ## Tokenix  x_train,x  tokenize_  ## Tokenix  x_train,x  tokenize_  ## Tokenix  x_train,x  tokenize_  ## Tokenix  and = conv.  in = Co	ConvID_with_filt  ConvID_with_filt  MaxPoolLayers  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  MaxPoolLayers  Flatten  MaxPoolLayers  Flatten  DropOut  Densel  OutputLa  Est,y_train,y_trai	## Size m: Conv  ## Size m: Conv  ## Size m: Conv  ## Size k: Conv  ## Size m: Conv  ## Siz	ID	### ### #### #### ####################	1.41. embeddings_initializer=tf.keras  1.41. embeddings_initializer=tf.keras  .initializers.he_normal(seed=42), kerne  s.initializers.he_normal(seed=42), kerne  s.initializers.he_normal(seed=42), kerne  s.initializers.he_normal(seed=42), kerne  s.initializers.he_normal(seed=42), kerne  cf.keras.initializers.he_normal(seed=42), kerne  cf.keras.initializers.
## Tokenized  ** Artain, x  ** Classifi  **	ConvID_with_filt  ConvID_with_filt  MaxPoolLayer1  ConvID_with_filt  ConvID_with_filt  MaxPoolLayer2  MaxPoolLayer2  Flatten  DropOul  Densel  OutputLa  Flatten  DropOul  Densel  OutputLa  Flatten  DropOul  Densel  OutputLa  Flatten  ConvID_with_filt  Flatten  DropOul  Densel  OutputLa  Flatten  ConvID_with_filt  Flatten  DropOul  Densel  OutputLa  Flatten  ConvID_with_filt  Flatten  DropOul  Densel  Densel  OutputLa  Flatten  ConvID_with_filt  Flatten  DropOul  Densel  Densel  OutputLa  Flatten  ConvID_with_filt  Flatten  DropOul  Densel  Densel  OutputLa  Flatten  DropOul  Densel  September = Token   100  God, 3, activation  10(64, 3, activation  11(64, 3, activation  12 = MaxPoolID(5) (1)  ID(64, 3, activation  13 = MaxPoolID(5) (1)  ID(64, 3, activation  14 = Dense (25, activation  15 = Dense (25, activation  16 = Dense (25, activation  17 = Dense (25, activation  18 = MaxPoolID(5) (1)  ID(64, 3, activation  19 = Dense (25, activation  10 = Dense (25, activation  10 = Dense (25, activation  11 = Dense (25, activation  12 = MaxPoolID(5) (1)  ID(64, 3, activation  13 = MaxPoolID(5) (1)  ID(64, 3, activation  14 = Dense (25, activation  15 = Dense (25, activation  16 = Dense (25, activation  17 = Dense (25, activation  18 = Dense (25, activation  19 = Dense (25, activation  10 = Dense (25, activation  10 = Dense (25, activation  11 = Dense (25, activation  12 = Dense (25, activation  13 = MaxPoolID(5) (1)  14 = Dense (25, activation  15 = Dense (25, activation  16 = Dense (25, activation  17 = Dense (25, activation  18 = Dense (25, activation  19 = Dense (25, activation  10 = Dense (2	resize_m: Conv  imaxPooling1D  resize_k: ConvI  resize_k:	ID	## ## ## ## ## ## ## ## ## ## ## ## ##	1.41. embeddings_initializer=tf.keras  1.41. embeddings_initializer=tf.keras  .initializers.he_normal(seed=42), kerne  s.initializers.he_normal(seed=42), kerne  s.initializers.he_normal(seed=42), kerne  s.initializers.he_normal(seed=42), kerne  s.initializers.he_normal(seed=42), kerne  cf.keras.initializers.he_normal(seed=42), kerne  cf.keras.initializers.
## Tokening  ** Atrain, x.  ** Classifi  **	ConvID_with_filt  ConvID_with_filt  MaxPoolLayer1  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  Flatten  DropOut  Densel  ConvID_with_filt  ConvID_ConvID_con	resize_m: Conv  i: MaxPoolingID  resize_k: ConvI  resize_t: ConvI  resize_	ID	## ## ## ## ## ## ## ## ## ## ## ## ##	Comment   Price   Pr
## Train, x  Tokenize, and in a control of a	ConvID_with_filt  ConvID_with_filt  MaxPoolLayer1  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  ConvID_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  ConvID_with_filt  ConvID_with_filt  MaxPoolLayer2  Flatten  DropOut  Densel  OutputLa  test,y_train,y_tr	resize_m: Conv  i: MaxPoolingID  resize_k: ConvI  resize_t: ConvI  resize_	ID	## ## ## ## ## ## ## ## ## ## ## ## ##	C. (A), entendings installingered Moras ( A), entendings installingered Moras ( A), installingers be permal (seeded2), bornel  A installingers be permal (se