Variables in Data Cleansing

X name	Data type	Description of each article	Y name	Data type
sci_full_texts_X	A list of 600 strings. Each string is an article.	Original article body	tags	A list of 6 strings. Each string is the category name. [‘biology’, …]
sci_string_X	(same as above)	(same as above)	index_to_tags_dict	A dict, length 6. {0: ‘biology’, 1: ‘chemistry’, …}
sci_string_char_free_X	A list of 600 strings. Each string is a processed article.	After deleting unwanted characters	tags_to_index_dict	A dict, length 6. {‘biology’: 0, ‘chemistry’: 1, …}
sci_full_contraction_X	A list of 600 strings. Each string is a processed article.	After lowercase and expanding contractions	sci_categories_target_Y	A list of 600 integers. Each integer is the category flag. In order (firstly 100 0’s, then 100 1’s, …).
sci_X_regularised	A list of 600 nested strings. Each big string is a processed article, containing many word strings.	After removing stopwords and lemmatisation		
sci_X_regularised_united	A list of 600 strings. Each string is a processed article.	After removing stopwords and lemmatisation		
sci_cleaned_combine	A list of 600 nested list. Each nested list contains the body text in string, and the integer flag [‘bodytext’, 3]		Combined X and Y	

Variables for Training and Testing, and for the Neural Network Models:

Variable name	Length	Its function	Original or Cleansed	Data type	Variable name	Length	Its function	Data type
X_train_pre	480	The training set split from the original sci_string_X	Original	A list of strings. Each string is an article.	Y_train_pre	480	The training set split from the sci_categories_target_Y	A list if integers. Each integer is the flag of that category.
X_test_pre	120	The test set split from the original sci_string_X	Original	A list of strings. Each string is an article.	Y_test_pre	120	The test set split from the sci_categories_target_Y	A list if integers. Each integer is the flag of that category.
X_train	480	The **Keras** accepted type of X_train_pre	Original	Nested numpy.ndarray. Each of the inner array is a one-hot encoding array which represents the article.	Y_train	480	The Keras accepted type of Y_train_pre	Nested numpy.ndarray. Each of the inner array is a one-hot encoding array which represents the category.
X_test	120	The **Keras** accepted type of X_test_pre	Original	Nested numpy.ndarray. Each of the inner array is a one-hot encoding array which represents the article.	Y_test	120	The Keras accepted type of Y_test_pre	Nested numpy.ndarray. Each of the inner array is a one-hot encoding array which represents the category.
X_train_pre2	480	The training set split from regularised sci_X_regularised_united	Cleansed	A list of strings. Each string is an article.	Y_train_pre2	480	The training set split from the sci_categories_target_Y	A list of integers ranging from 0 to 5
X_test_pre2	120	The test set split from regularised sci_X_regularised_united	Cleansed	A list of strings. Each string is an article.	Y_test_pre2	120	The test set split from the sci_categories_target_Y	A list of integers ranging from 0 to 5
X_train2	480	The **Keras** accepted type of X_train_pre2	Cleansed	numpy.ndarray [[0, 1, 1,...], [1, 0, 1, ...]]	Y_train2	480	The Keras accepted type of Y_train_pre2	numpy.ndarray Each one is a one-hot encoding like [1., 0., 0., 0., 0., 0.]
X_test2	120	The **Keras** accepted type of X_test_pre2	Cleansed	numpy.ndarray [[0, 1, 1,...], [1, 0, 1, ...]]	Y_test2	120	The Keras accepted type of Y_test_pre2	numpy.ndarray Each one is a one-hot encoding like [1., 0., 0., 0., 0., 0.]

Experiments Records – batch size

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	**Batch size**	# hidden layers	Neuron nums	Activation functions	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Dec 09	Cleansed	15	15	15	600	20	**1024**	1	64-64-6	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.5667	1.1342
Dec 09	Cleansed	15	15	15	600	20	**512**	1	64-64-6	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.5750	1.1667
Dec 09	Cleansed	15	15	15	600	20	**256**	1	64-64-6	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6250	1.0836
Dec 09	Cleansed	15	15	15	600	20	**128**	1	64-64-6	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6333	1.0605
Dec 09	Cleansed	15	15	15	600	20	**64**	1	64-64-6	relu-relu-softmax	rmsprop	categorical_crossentropy	-	**0.6750**	1.1273
Dec 09	Cleansed	15	15	15	600	20	**32**	1	64-64-6	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6250	1.2453
Dec 09	Cleansed	15	15	15	600	20	**16**	1	64-64-6	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.5500	1.3784

Experiments Records – number of layers

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	**# hidden layers**	Neuron nums	Activation functions	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Jan 31	Cleansed	15	15	15	600	20	**64**	**0**	**256-6**	relu-softmax	rmsprop	categorical_crossentropy	-	0.6583333611488342	1.028511643409729
Jan 31	Cleansed	15	15	15	600	20	**64**	**1**	**256-256-6**	relu-relu-softmax	rmsprop	categorical_crossentropy	-	**0.6916666626930237**	1.1270121335983276
Jan 31	Cleansed	15	15	15	600	20	**64**	**2**	**256-256-256-6**	relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6416666507720947	1.2906997203826904
Jan 31	Cleansed	15	15	15	600	20	**64**	**3**	**256-256-256-256-6**	relu-relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6499999761581421	1.201580286026001
Jan 31	Cleansed	15	15	15	600	20	**64**	**4**	**256-256-256-256-256-6**	relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.675000011920929	1.114376425743103
Jan 31	Cleansed	15	15	15	600	20	**64**	**5**	**256-256-256-256-256-256-6**	relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.675000011920929	1.5121153593063354
Jan 31	Cleansed	15	15	15	600	20	**64**	**6**	**256-256-256-256-256-256-256-256-6**	relu- relu- relu-relu-relu-relu-relu -softmax	rmsprop	categorical_crossentropy	-	0.6416666507720947	1.577928900718689

Experiments Records – number of layers

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	**# hidden layers**	Neuron nums	Activation functions	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Jan 31	Cleansed	15	15	15	600	20	**256**	**0**	**128-6**	relu-softmax	rmsprop	categorical_crossentropy	-	0.675000011920929	0.9862571954727173
Jan 31	Cleansed	15	15	15	600	20	**256**	**1**	**128-128-6**	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6833333373069763	0.9424988031387329
Jan 31	Cleansed	15	15	15	600	20	**256**	**2**	**128-128-128-6**	relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.625	1.1201564073562622
Jan 31	Cleansed	15	15	15	600	20	**256**	**3**	**128-128-128-128-6**	relu-relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	**0.699999988079071**	1.0144932270050049
Jan 31	Cleansed	15	15	15	600	20	**256**	**4**	**128-128-128-128-128-6**	relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.5833333134651184	1.1369404792785645
Jan 31	Cleansed	15	15	15	600	20	**256**	**5**	**128-128-128-128-128-128-6**	relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.6083333492279053	1.069790244102478
Jan 31	Cleansed	15	15	15	600	20	**256**	**6**	**128-128-128-128-128-128-128-6**	relu- relu- relu-relu-relu-relu-relu -softmax	rmsprop	categorical_crossentropy	-	0.625	1.1483782529830933

Experiments Records – number of layers

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	**# hidden layers**	Neuron nums	Activation functions	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Jan 31	Cleansed	15	15	15	600	20	**128**	**0**	**128-6**	relu-softmax	rmsprop	categorical_crossentropy	-	**0.699999988079071**	0.9842715859413147
Jan 31	Cleansed	15	15	15	600	20	**128**	**1**	**128-128-6**	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6583333611488342	1.0232696533203125
Jan 31	Cleansed	15	15	15	600	20	**128**	**2**	**128-128-128-6**	relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	**0.699999988079071**	0.9746896624565125
Jan 31	Cleansed	15	15	15	600	20	**128**	**3**	**128-128-128-128-6**	relu-relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6416666507720947	1.1576184034347534
Jan 31	Cleansed	15	15	15	600	20	**128**	**4**	**128-128-128-128-128-6**	relu-relu-relu-relu-relu -softmax	rmsprop	categorical_crossentropy	-	0.6000000238418579	1.2234609127044678
Jan 31	Cleansed	15	15	15	600	20	**128**	**5**	**128-128-128-128-128-128-6**	relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.5916666388511658	1.2051795721054077
Jan 31	Cleansed	15	15	15	600	20	**128**	**6**	**128-128-128-128-128-128-128-128-6**	relu- relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.6333333253860474	1.111493706703186

Experiments Records – number of layers

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	**# hidden layers**	Neuron nums	Activation functions	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Feb 03	Cleansed	15	15	15	600	20	**64**	**0**	**256-6**	relu-softmax	rmsprop	categorical_crossentropy	-	**0.7333333492279053**	0.9777682423591614
Feb 03	Cleansed	15	15	15	600	20	**64**	**1**	**256-256-6**	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6833333373069763	1.1629196405410767
Feb 03	Cleansed	15	15	15	600	20	**64**	**2**	**256-256-256-6**	relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6833333373069763	1.229711651802063
Feb 03	Cleansed	15	15	15	600	20	**64**	**3**	**256-256-256-256-6**	relu-relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6083333492279053	1.6497364044189453
Feb 03	Cleansed	15	15	15	600	20	**64**	**4**	**256-256-256-256-256-6**	relu-relu-relu-relu-relu -softmax	rmsprop	categorical_crossentropy	-	0.6166666746139526	1.4505549669265747
Feb 03	Cleansed	15	15	15	600	20	**64**	**5**	**256-256-256-256-256-256-6**	relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.6666666865348816	1.3347638845443726
Feb 03	Cleansed	15	15	15	600	20	**64**	**6**	**256-256-256-256-256-256-256-6**	relu- relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.6583333611488342	1.6978732347488403

Experiments Records – number of layers

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	**# hidden layers**	Neuron nums	Activation functions	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Feb 03	Cleansed	15	15	15	600	20	**256**	**0**	**128-6**	relu-softmax	rmsprop	categorical_crossentropy	-	0.6166666746139526	1.0157369375228882
Feb 03	Cleansed	15	15	15	600	20	**256**	**1**	**128-128-6**	relu-relu-softmax	rmsprop	categorical_crossentropy	-	**0.6666666865348816**	0.98311448097229
Feb 03	Cleansed	15	15	15	600	20	**256**	**2**	**128-128-128-6**	relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6583333611488342	1.1143690347671509
Feb 03	Cleansed	15	15	15	600	20	**256**	**3**	**128-128-128-128-6**	relu-relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.5916666388511658	1.1187578439712524
Feb 03	Cleansed	15	15	15	600	20	**256**	**4**	**128-128-128-128-128-6**	relu-relu-relu-relu-relu -softmax	rmsprop	categorical_crossentropy	-	**0.6666666865348816**	0.9662565588951111
Feb 03	Cleansed	15	15	15	600	20	**256**	**5**	**128-128-128-128-128-128-6**	relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.6000000238418579	1.0963152647018433
Feb 03	Cleansed	15	15	15	600	20	**256**	**6**	**128-128-128-128-128-128-128-128-6**	relu- relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.5083333253860474	1.6235156059265137

Experiments Records – number of layers

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	**# hidden layers**	Neuron nums	Activation functions	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Feb 03	Cleansed	15	15	15	600	20	**128**	**0**	**128-6**	relu-softmax	rmsprop	categorical_crossentropy	-	**0.675000011920929**	0.9399003386497498
Feb 03	Cleansed	15	15	15	600	20	**128**	**1**	**128-128-6**	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6499999761581421	1.0389785766601562
Feb 03	Cleansed	15	15	15	600	20	**128**	**2**	**128-128-128-6**	relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6333333253860474	1.212424397468567
Feb 03	Cleansed	15	15	15	600	20	**128**	**3**	**128-128-128-128-6**	relu-relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6499999761581421	1.1811801195144653
Feb 03	Cleansed	15	15	15	600	20	**128**	**4**	**128-128-128-128-128-6**	relu-relu-relu-relu-relu -softmax	rmsprop	categorical_crossentropy	-	0.625	1.2557777166366577
Feb 03	Cleansed	15	15	15	600	20	**128**	**5**	**128-128-128-128-128-128-6**	relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.6000000238418579	1.373138427734375
Feb 03	Cleansed	15	15	15	600	20	**128**	**6**	**128-128-128-128-128-128-128-128-6**	relu- relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.5	1.9701672792434692

Experiments Records – activation functions

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	# hidden layers	Neuron nums	**Activation functions**	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-relu-softmax**	rmsprop	categorical_crossentropy	-	0.65	1.0465116818745932
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-softmax-softmax**	rmsprop	categorical_crossentropy	-	0.49166667	1.6607657194137573
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-sigmoid-softmax**	rmsprop	categorical_crossentropy	-	0.675	0.9671117067337036
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-tanh-softmax**	rmsprop	categorical_crossentropy	-	0.6333333	1.1139755964279174
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-selu-softmax**	rmsprop	categorical_crossentropy	-	0.6666667	1.0172423601150513
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-softsign-softmax**	rmsprop	categorical_crossentropy	-	0.68333334	0.9269145607948304
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-hard_sigmoid-softmax**	rmsprop	categorical_crossentropy	-	**0.69166666**	0.9540115276972453
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-exponential-softmax**	rmsprop	categorical_crossentropy	-	0.6666667	1.4939833 08474223

Experiments Records – activation functions

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	# hidden layers	Neuron nums	**Activation functions**	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Feb 03	Cleansed	15	15	15	600	20	128	2	64-128-128-6	**relu-relu-relu-softmax**	rmsprop	categorical_crossentropy	-	0.625	1.0755352258682251
Feb 03	Cleansed	15	15	15	600	20	128	2	64-128-128-6	**relu-softmax-softmax-softmax**	rmsprop	categorical_crossentropy	-	0.29166666	1.7877373139063517
Feb 03	Cleansed	15	15	15	600	20	128	2	64-128-128-6	**relu-sigmoid-sigmoid-softmax**	rmsprop	categorical_crossentropy	-	0.6166667	1.104865050315857
Feb 03	Cleansed	15	15	15	600	20	128	2	64-128-128-6	**relu-tanh-tanh-softmax**	rmsprop	categorical_crossentropy	-	**0.6666667**	1.0579676866531371
Feb 03	Cleansed	15	15	15	600	20	128	2	64-128-128-6	**relu-selu-selu-softmax**	rmsprop	categorical_crossentropy	-	0.64166665	1.2318978706995647
Feb 03	Cleansed	15	15	15	600	20	128	2	64-128-128-6	**relu-softsign-softsign-softmax**	rmsprop	categorical_crossentropy	-	0.625	1.1883386691411337
Feb 03	Cleansed	15	15	15	600	20	128	2	64-128-128-6	**relu—hard_sigmoid-hard_sigmoid-softmax**	rmsprop	categorical_crossentropy	-	0.6333333	1.1228162209192911
Feb 03	Cleansed	15	15	15	600	20	128	2	64-128-128-6	**relu-exponential-exponential-softmax**	rmsprop	categorical_crossentropy	-	0.55833334	1.981887149810791

Experiments Records – neuron numbers

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	# hidden layers	**Neuron nums**	Activation functions	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Feb 03	Cleansed	15	15	15	600	15	512	2	**32-32-32-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.575	1.281682046254476
Feb 03	Cleansed	15	15	15	600	15	512	2	**32-64-64-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.5416667	1.2656379699707032
Feb 03	Cleansed	15	15	15	600	15	512	2	**64-64-64-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.625	1.1398384332656861
Feb 03	Cleansed	15	15	15	600	15	512	2	**64-64-32-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.55	1.1970116376876831
Feb 03	Cleansed	15	15	15	600	15	512	2	**64-128-128-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.55	1.1849392811457315
Feb 03	Cleansed	15	15	15	600	15	512	2	**64-128-64-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.55833334	1.1169984579086303
Feb 03	Cleansed	15	15	15	600	15	512	2	**128-128-256-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	**0.68333334**	0.9293365836143493
Feb 03	Cleansed	15	15	15	600	15	512	2	**128-128-128-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.53333336	1.320115900039673
Feb 03	Cleansed	15	15	15	600	15	512	2	**128-128-64-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.6	1.1395802736282348
Feb 03	Cleansed	15	15	15	600	15	512	2	**128-256-128-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.6166667	1.1106221596399943

Feb 03	Cleansed	15	15	15	600	15	512	2	**128-256-64-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.56666666	1.1338408788045247
Feb 03	Cleansed	15	15	15	600	15	512	2	**512-256-128-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	**0.68333334**	0.9334465821584066
Feb 03	Cleansed	15	15	15	600	15	512	2	**512-512-256-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.65833336	0.9807055513064067
Feb 03	Cleansed	15	15	15	600	15	512	2	**512-1024-512-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.53333336	1.1604699532190959
Feb 03	Cleansed	15	15	15	600	15	512	2	**1024-1024-512-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.65833336	0.9688995122909546
Feb 03	Cleansed	15	15	15	600	15	512	2	**1024-512-256-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.55	1.2143640756607055
Feb 03	Cleansed	15	15	15	600	15	512	2	**1024-512-128-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.55833334	1.0940533717473349

Experiments Records – optimizers

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	# hidden layers	Neuron nums	Activation functions	**optimizer**	loss	regularisation	Accuracy (test)	Loss (test)
Feb 03	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	**rmsprop**	categorical_crossentropy	-	0.65833336	1.137278159459432
Feb 03	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	**adam**	categorical_crossentropy	-	**0.7083333**	1.2403667132059732
Feb 03	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	**sgd**	categorical_crossentropy	-	0.425	1.4351878404617309
Feb 03	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	**adagrad**	categorical_crossentropy	-	0.35833332	1.663960321744283
Feb 03	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	**adamax**	categorical_crossentropy	-	0.69166666	1.0454886635144551
Feb 03	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	**adadelta**	categorical_crossentropy	-	0.175	1.8860424121220907

Experiments Records – regularisation

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	# hidden layers	Neuron nums	Activation functions	optimizer	loss	**Regulerisation (same for all 4 layers)**	Accuracy (test)	Loss (test)
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**None**	0.6583333611488342	1.2913317680358887
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1(0.00001)**	0.6666666865348816	1.4933713674545288
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1(0.0001)**	0.6833333373069763	2.95630884170532 23
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1(0.001)**	0.65833336114883 42	7.496500015258789
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1(0.01)**	0.16666667163372 04	20.8071060180664 06
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1(0.1)**	0.20833332836627 96	186.671417236328 12
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l2(0.00001)**	**0.69166666269302 37**	1.30076277256011 96
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l2(0.0001)**	0.68333333730697 63	1.41088056564331 05

Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l2(0.001)**	0.66666668653488 16	2.10592889785766 6
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l2(0.01)**	**0.69166666269302 37**	3.84803867340087 9
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l2(0.1)**	0.44166666269302 37	10.2590351104736 33
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1_l2(l1=0.00001, l2=0.00001)**	0.64999997615814 21	1.47787427902221 68
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1_l2(l1=0.0001, l2=0.0001)**	0.65833336114883 42	2.92774510383605 96
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1_l2(l1=0.001, l2=0.001)**	**0.69166666269302 37**	7.60630893707275 4
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1_l2(l1=0.1, l2=0.1)**	0.19166666269302 368	200.353195190429 7
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1_l2(l1=0.00001, l2=0.1)**	0.47499999403953 55	10.3081378936767 58
Feb 08	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	**l1_l2(l1=0.1, l2=0.00001)**	0.19166666269302 368	186.403411865234 38

Repetition of High Accuracy Experiments

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	# hidden layers	Neuron nums	Activation functions	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-		
	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adam	categorical_crossentropy	-		
	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	sgd	categorical_crossentropy	-		
	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adagrad	categorical_crossentropy	-		
	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adamax	categorical_crossentropy	-		
	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	adadelta	categorical_crossentropy	-		

Controlled Experiments Records – Summary of Relatively High Accuracy Models

(In each row, the cell with red bold characters is the independent variable.)

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	# hidden layers	Neuron nums	Activation functions	optimizer	loss	regularisation	Accuracy (test)	Loss (test)
Feb 03	Cleansed	15	15	15	600	20	**64**	**0**	**256-6**	relu-softmax	rmsprop	categorical_crossentropy	-	0.7333333492279053	0.9777682423591614
Feb 03	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	**adam**	categorical_crossentropy	-	0.7083333	1.2403667132059732
Jan 31	Cleansed	15	15	15	600	20	**256**	**3**	**128-128-128-128-6**	relu-relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.699999988079071	1.0144932270050049
Jan 31	Cleansed	15	15	15	600	20	**128**	**0**	**128-6**	relu-softmax	rmsprop	categorical_crossentropy	-	0.699999988079071	0.9842715859413147
Jan 31	Cleansed	15	15	15	600	20	**128**	**2**	**128-128-128-6**	relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.699999988079071	0.9746896624565125
Jan 31	Cleansed	15	15	15	600	20	**64**	**1**	**256-256-6**	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6916666626930237	1.1270121335983276
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-hard_sigmoid-softmax**	rmsprop	categorical_crossentropy	-	0.69166666	0.9540115276972453
Feb 03	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	**adamax**	categorical_crossentropy	-	0.69166666	1.0454886635144551
Jan 31	Cleansed	15	15	15	600	20	**256**	**1**	**128-128-6**	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6833333373069763	0.9424988031387329
Feb 03	Cleansed	15	15	15	600	20	**64**	**1**	**256-256-6**	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6833333373069763	1.1629196405410767

Feb 03	Cleansed	15	15	15	600	20	**64**	**2**	**256-256-256-6**	relu- relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6833333373069763	1.229711651802063
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-softsign-softmax**	rmsprop	categorical_crossentropy	-	0.68333334	0.9269145607948304
Feb 03	Cleansed	15	15	15	600	15	512	2	**128-128-256-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.68333334	0.9293365836143493
Feb 03	Cleansed	15	15	15	600	15	512	2	**512-256-128-6**	relu-tanh-tanh-softmax	rmsprop	categorical_crossentropy	-	0.68333334	0.9334465821584066
Dec 09	Cleansed	15	15	15	600	20	**64**	1	64-64-6	relu-relu-softmax	rmsprop	categorical_crossentropy	-	0.6750	1.1273
Jan 31	Cleansed	15	15	15	600	20	**64**	**4**	**256-256-256-256-256-6**	relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.675000011920929	1.114376425743103
Jan 31	Cleansed	15	15	15	600	20	**64**	**5**	**256-256-256-256-256-256-256-6**	relu- relu-relu-relu-relu- relu -softmax	rmsprop	categorical_crossentropy	-	0.675000011920929	1.5121153593063354
Jan 31	Cleansed	15	15	15	600	20	**256**	**0**	**128-6**	relu-softmax	rmsprop	categorical_crossentropy	-	0.675000011920929	0.9862571954727173
Feb 03	Cleansed	15	15	15	600	20	**128**	**0**	**128-6**	relu-softmax	rmsprop	categorical_crossentropy	-	0.675000011920929	0.9399003386497498
Feb 03	Cleansed	15	15	15	600	20	128	1	64-128-6	**relu-sigmoid-softmax**	rmsprop	categorical_crossentropy	-	0.675	0.9671117067337036

Table 5.1.1-2 Repetition for the High-Performing FNN Models Using K-Fold Cross Validation

Settings						Tuning parameters								Evaluations	
Date	Data	sys_seed	np_seed	tf_seed	words	Epochs	Batch size	# hid layers	Neuron nums	Activation functions	optimizer	loss	regulerisation	Accuracy Holdout (test)	Avg Accuracy 10-Fold (val)
Feb 28	Cleansed	15	15	15	600	20	**64**	**0**	**256-6**	relu-softmax	rmsprop	categorical_crossentropy	-	0.7083333134651184	60.63% (+/-4.79%)
Feb 23	Cleansed	15	15	15	600	20	**64**	**0**	**256-6**	relu-softmax	rmsprop	categorical_crossentropy	Dropout(0.3)	0.7083333134651184	63.54% (+/-6.67%)
Feb 28	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	**adam**	categorical_crossentropy	-	0.6833333373069763	60.21% (+/-5.70%)
Feb 28	Cleansed	15	15	15	600	20	128	2	512-256-128-6	relu-tanh-tanh-softmax	**adam**	categorical_crossentropy	2 Dropout	-	61.04% (+/-6.85%)
Feb 28	Cleansed	15	15	15	600	20	**256**	**3**	**128-128-128-128-6**	relu-relu-relu-relu-softmax	rmsprop	categorical_crossentropy	3 Dropout	0.699999988079071	59.38% (+/-6.86%)

Grid Search Results

Feb 02 03:00

hyperparameters2 = {

'neuron_num1': [32, 64],

'neuron_num2': [128, 256, 512],

'act_func': ['relu', 'tanh', 'sigmoid', 'selu'],

'batch_size': [64, 128, 256, 512],

'epochs': [20, 40, 70],

'loss_f': ['categorical_crossentropy', 'mean_absolute_error'],

'optimizer': ['adam', 'rmsprop', 'sgd', 'adagrad']

}

```
Best hyperparameters are: {'act_func': 'tanh', 'batch_size': 256, 'epochs': 20, 'loss_f': 'mean_abs
olute_error', 'neuron_num1': 64, 'neuron_num2': 128, 'optimizer': 'rmsprop'} Best score is:  0.5520
833333333333
```

In [ ]:

Feb 03 4:55

hyperparameters3 = {

'neuron_num1': [32, 64],

'neuron_num2': [128, 256, 512],

'num_layers': [0, 1, 2, 3],

'act_func': ['relu', 'tanh', 'selu'],

'batch_size': [128, 256, 512],

'epochs': [20, 40, 60],

'loss_f': ['categorical_crossentropy', 'mean_absolute_error'],

'optimizer': ['adam', 'rmsprop', 'adamax']

}

```
Best hyperparameters are: {'act_func': 'relu', 'batch_size': 256, 'epochs': 60, 'loss_f': 'mean_absolute_error', '
neuron_num1': 64, 'neuron_num2': 256, 'num_layers': 0, 'optimizer': 'adam'} Best score is:  0.5625
```

Methods other than pure feed-forward neural networks:

Settings						Transfer Learning Source	Layer Settings				Evaluations		
Date	Data	Model name	np_seed	tf_seed	words		Layers	Details	Batch size	Epoch	Accuracy (val)	Accuracy (test)	Loss (test)
Dec 09	Cleansed	1dCNN	15	15	600	Chollet's Ch. 6.4	Embedding, Conv1D, MaxPooling1D, Conv1D, GlobalMaxPooling1D, Dense	layers.Embedding(10000, 128, input_length=600) layers.Conv1D(32, 6, activation='relu') layers.MaxPooling1D(15) layers.Conv1D(32, 6, activation='relu') layers.GlobalMaxPooling1D() layers.Dense(6)	64	25	0.2188	0.125	9.1336
						Chollet's Ch. 6.4	(above)	(above)	128	1	0.2188	0.1417	5.1839
Feb 03	Cleansed	1dCNN	15	15	600	https://www.bilibili.com/video/BV1u7411d7zU/?share_source=copy_web&vd	Embedding, Conv1D, MaxPooling1D, Conv1D, Flatten, Dropout, BatchNormal	model_1DCNN.add(layers.Embedding(input_dim=2000, output_dim=128, input_length=600)) model_1DCNN.add(layers.Conv1D(256, 3, padding='same', activation='relu'))	64	25	0.5208	**0.57499998807907 1**	1.27823185920715 33

						_source=296c14837e03501f00801a512d70f87e	ization, Dense, Dropout, Dense	model_1DCNN.add(layers.MaxPooling1D(3, 3, padding='same')) model_1DCNN.add(layers.Conv1D(32, 3, padding='same', activation='relu')) model_1DCNN.add(layers.Flatten()) model_1DCNN.add(layers.Dropout(0.3)) model_1DCNN.add(layers.BatchNormalization()) model_1DCNN.add(layers.Dense(256, activation='relu')) model_1DCNN.add(layers.Dropout(0.2)) model_1DCNN.add(layers.Dense(6, activation='softmax'))					
Feb 05	Cleansed	1dCNN	15	15	600	the book 'Python 深度学习' Section 4.7	Embedding, Cov1D, MaxPooling1D, Conv1D, Flatten, Dropout, BatchNormalization, Dense, Dropout, Sense	model_CNN02.add(Embedding(input_dim = 600,output_dim = 32,input_length=600)) model_CNN02.add(Conv1D(256, 3, padding='same', activation='relu')) model_CNN02.add(MaxPooling1D(3, 3, padding='same')) model_CNN02.add(Conv1D(32, 3, padding='same', activation='relu')) model_CNN02.add(Flatten()) model_CNN02.add(Dropout(0.3)) model_CNN02.add(BatchNormalizati	256	15	0.1979	0.316666662693023 7	1.62513089179992 68

								on()) model_CNN02.add(Dense(256, activation='relu')) model_CNN02.add(Dropout(0.2)) model_CNN02.add(Dense(6, activation='softmax')) model_CNN02.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])					
Feb 03	Cleansed	LSTM layer	15	15	600	Chollet's book Ch.6	Embedding, LSTM, Dropout, LSTM, Dropout, Dense	model_RNN.add(Embedding(600, 128)) model_RNN.add(LSTM(64, activation='tanh', ...)) model_RNN.add(Dropout(0.4)) model_RNN.add(LSTM(64, activation='tanh')) model_RNN.add(Dropout(0.4)) model_RNN.add(Dense(6, activation='softmax'))	128	15	0.2604	0.266666680574417 1	1.73751449584960 94
Feb 04	Cleansed	RNN	15	15	600	Chollet's Ch. 6.4	Embedding, SimpleRNN, Dense	model_RNN01.add(Embedding(10000, 128, input_length=600)) model_RNN01.add(SimpleRNN(128)) model_RNN01.add(Dense(6, activation='sigmoid')) model_RNN01.compile(optimizer='rmsprop', loss='categorical_crossentropy',	64	25	0.3125	0.258333325386047 36	1.78740382194519 04

								metrics=['accuracy'])					
Feb 04	Cleansed	RNN	15	15	600	Book - TensorFlow Machine Learning Cookbook, by Nick McClure		(bugs and errors)					
Feb 04	Cleansed	Fune-tuned AWD-LSTM	15	15	600	Merity et al., github.com/salesforce/awd-lstm-lm		sgd = SGD(learning_rate=1, weight_decay=0.0000012) model_RNN.add(Embedding(600, 400)) model_RNN.add(Dropout(0.1)) model_RNN.add(LSTM(200, activation='tanh', dropout=0.0, recurrent_dropout=0.0, return_sequences=True)) model_RNN.add(Dropout(0.4)) model_RNN.add(SimpleRNN(200)) model_RNN.add(Dropout(0.2)) model_RNN.add(Dense(6, activation='softmax')) model_RNN.compile(optimizer=sgd, loss='categorical_crossentropy', metrics=['accuracy'])	80	15	0.1771	0.166666671633720 4	3071.2744140625
Feb 04	Cleansed	Bidirectional LSTM layer	15	15	600	the book TensorFlow 从零开始学	Embedding, Bidirectional LSTM, Dropout, Dense,	model_RNN04.add(Embedding(600, 16)) model_RNN04.add(Bidirectional(LST	128	40	0.3229	0.25	1.68488943576812 74

						Section 7.3	Dense	M(16))) model_RNN04.add(Dropout(0.3)) model_RNN04.add(Dense(16, activation='relu')) model_RNN04.add(Dense(6, activation='softmax')) model_RNN04.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])					

The K-fold Cross Validation Results for the Models Listed in the Previous Table

Note: the random state for k-fold splitting is 15.

Settings						Transfer Learning Source	Layer Settings				K-fold?	Evaluation
Date	Data	Model name	np_seed	tf_seed	words		Layers	Details	Batch size	Epoch		Avg accuracy (val)
Feb 28	Cleansed	1dCNN	15	15	600	Chollet's Ch. 6.4	Embedding, Conv1D, MaxPooling1D, Conv1D, GlobalMaxPooling1D, Dense	layers.Embedding(10000, 128, input_length=600) layers.Conv1D(32, 6, activation='relu') layers.MaxPooling1D(15) layers.Conv1D(32, 6, activation='relu') layers.GlobalMaxPooling1D() layers.Dense(6)	64	25	10-fold, rand=15	16.67% (+/-1.86%)
Feb 28	Cleansed	1dCNN	15	15	600	https://www.bilibili.com/video/BV1u7411d7zU/?share_source=copy_web&vd_source=296c14837e03501f00801a512d70f87e	Embedding, Conv1D, MaxPooling1D, Conv1D, Flatten, Dropout, BatchNormalization, Dense, Dropout, Dense	model_1DCNN.add(layers.Embedding(input_dim=2000, output_dim=128, input_length=600)) model_1DCNN.add(layers.Conv1D(256, 3, padding='same', activation='relu')) model_1DCNN.add(layers.MaxPooling1D(3, 3, padding='same')) model_1DCNN.add(layers.Conv1D(32, 3, padding='same', activation='relu')) model_1DCNN.add(layers.Flatten()) model_1DCNN.add(layers.Dropout(0.3))	64	25	10-fold, rand=15	54.37% (+/-3.54%)

								model_1DCNN.add(layers.BatchNormalization()) model_1DCNN.add(layers.Dense(256, activation='relu')) model_1DCNN.add(layers.Dropout(0.2)) model_1DCNN.add(layers.Dense(6, activation='softmax'))				
Feb 28	Cleansed	1dCNN	15	15	600	the book 'Python 深度学习' Section 4.7	Embedding, Cov1D, MaxPooling1D, Conv1D, Flatten, Dropout, BatchNormalization, Dense, Dropout, Sense	model_CNN02.add(Embedding(input_dim = 600,output_dim = 32,input_length=600)) model_CNN02.add(Conv1D(256, 3, padding='same', activation='relu')) model_CNN02.add(MaxPooling1D(3, 3, padding='same')) model_CNN02.add(Conv1D(32, 3, padding='same', activation='relu')) model_CNN02.add(Flatten()) model_CNN02.add(Dropout(0.3)) model_CNN02.add(BatchNormalization()) model_CNN02.add(Dense(256, activation='relu')) model_CNN02.add(Dropout(0.2)) model_CNN02.add(Dense(6, activation='softmax')) model_CNN02.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])	256	15	10-fold, rand=15	31.25% (+/-9.13%)
	Cleanse	LSTM layer	15	15	600	Chollet's book Ch.6	Embedding, LSTM,	model_RNN.add(Embedding(600, 128)) model_RNN.add(LSTM(64,	128	15		

	d						Dropout, LSTM, Dropout, Dense	activation='tanh', ...)) model_RNN.add(Dropout(0.4)) model_RNN.add(LSTM(64, activation='tanh')) model_RNN.add(Dropout(0.4)) model_RNN.add(Dense(6, activation='softmax'))				
	Cleansed	RNN	15	15	600	Chollet's Ch. 6.4	Embedding, SimpleRNN, Dense	model_RNN01.add(Embedding(10000, 128, input_length=600)) model_RNN01.add(SimpleRNN(128)) model_RNN01.add(Dense(6, activation='sigmoid')) model_RNN01.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy'])	64	25		
	Cleansed	RNN	15	15	600	Book - TensorFlow Machine Learning Cookbook, by Nick McClure		(bugs and errors)				
	Cleansed	Fune-tuned AWD-LSTM	15	15	600	Merity et al., github.com/salesforce/awd-lstm-lm		sgd = SGD(learning_rate=1, weight_decay=0.0000012) model_RNN.add(Embedding(600, 400)) model_RNN.add(Dropout(0.1)) model_RNN.add(LSTM(200, activation='tanh', dropout=0.0,	80	15		

								recurrent_dropout=0.0, return_sequences=True)) model_RNN.add(Dropout(0.4)) model_RNN.add(SimpleRNN(200)) model_RNN.add(Dropout(0.2)) model_RNN.add(Dense(6, activation='softmax')) model_RNN.compile(optimizer=sgd, loss='categorical_crossentropy', metrics=['accuracy'])				
	Cleansed	Bidirectional LSTM layer	15	15	600	the book TensorFlow 从零开始学 Section 7.3	Embedding, Bidirectional LSTM, Dropout, Dense, Dense	model_RNN04.add(Embedding(600, 16)) model_RNN04.add(Bidirectional(LSTM(16))) model_RNN04.add(Dropout(0.3)) model_RNN04.add(Dense(16, activation='relu')) model_RNN04.add(Dense(6, activation='softmax')) model_RNN04.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])	128	40		

Fine Tuning for the best performing CNN model:

Settings						Transfer Learning Source	Layer Settings				Evaluations			k-fold
Date	Data	Model name	np_seed	tf_seed	words		Layers	Details	Batch size	Epoch	Accuracy (val)	Accuracy (test)	Loss (test)	Avg Accuracy using 10-fold
Feb 03	Cleansed	1dCNN	15	15	600	https://www.bilibili.com/video/BV1u7411d7zU/?share_source=copy_web&vd_source=296c14837e03501f00801a512d70f87e	Embedding, Conv1D, MaxPooling1D, Conv1D, Flatten, Dropout, BatchNormalization, Dense, Dropout, Dense	layers.Embedding(input_dim=2000, output_dim=128, input_length=600) layers.Conv1D(256, 3, padding='same', activation='relu') layers.MaxPooling1D(3, 3, padding='same') layers.Conv1D(32, 3, padding='same', activation='relu') layers.Flatten() layers.Dropout(0.3) layers.BatchNormalization() layers.Dense(256, activation='relu') layers.Dropout(0.2) layers.Dense(6, activation='softmax') optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy']	64	25	0.5208	**0.574999988079071**	1.27823185920715 33	51.25% (+/- 3.51%)

Feb 08	same	same	same	same	same	same	same	1st Embeddin, input_dim=1800	256	30	0.5208	0.5583333373069763	1.4914071559906006	
Feb 08	same	same	same	same	same	same	same	1st Embeddin, input_dim=3000	256	30	0.4688	0.5249999761581421	1.4622343778610223	
Feb 08	same	same	same	same	same	same	same	1st Embeddin, input_dim=500	256	28	0.4688	0.5249999761581421	1.4969735145568848	
Feb 08	same	same	same	same	same	same	same	same	same	28	0.6146	**0.5916666388511658**	1.2260761260986328	
Feb 08	same	same	same	same	same	same	same	same	same	35	0.6250 (overfitting)	0.5416666865348816	1.1961406469345093	
Feb 08	same	same	same	same	same	same	same	same	same	30	0.6146	**0.625**	1.2466410398483276	
Feb 08	same	same	same	same	same	same	same	same	same	31	0.5417	**0.63333332538604**	1.3252640962	53.12% (+/-4.39%)

												74	600708	
Feb 08	same	same	same	same	same	same	2 set of MaxPooling1D, Conv1D, Dropout. Flatten moved.	layers.MaxPooling1D(3, 3, padding='same') layers.Conv1D(64, 3, padding='same', activation='relu') layers.Dropout(0.3) **layers.MaxPooling1D(3, 3, padding='same')** **layers.Conv1D(32, 2, padding='same', activation='relu')** **layers.Flatten()** **layers.Dropout(0.3)** layers.BatchNormalization() layers.Dense(256, activation='relu') layers.Dropout(0.2) layers.Dense(6, activation='softmax') optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy']	same	40	0.5833	0.5249999761581421	1.2528330087661743	
Feb 08	same	same	same	same	same	same	2 set of MaxPooling1D, Conv1D, Dropout. Flatten moved. Using	layers.MaxPooling1D(3, 3, padding='same') layers.Conv1D(64, 3, padding='same', activation='relu') layers.Dropout(0.2) layers.MaxPooling1D(3, 3,	same	40	0.5521	0.574999988079071	1.4414136409759521	

							tanh.	padding='same') layers.Conv1D(32, 2, padding='same', activation='tanh') layers.Flatten() layers.Dropout(0.2) layers.BatchNormalization()) # the layer of batch normalization layers.Dense(128, activation='tanh') layers.Dropout(0.2) layers.Dense(64, activation='tanh') layers.Dropout(0.1) layers.Dense(6, activation='softmax') optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy']						
Feb 08	Clean sed	1dCNN	15	15	600			layers.Embedding(input_dim=2000, output_dim=128, input_length=600) layers.Conv1D(256, 3, padding='same', activation='relu') layers.MaxPooling1D(3, 3, padding='same') layers.Conv1D(32, 2, padding='same', activation='tanh') layers.Flatten() layers.Dropout(0.2) layers.BatchNormalization()	64	40	0.57 29	**0.6166 66674 61395 26**	1.40 9921 5269 0887 45	

								layers.Dense(128, activation='tanh') layers.Dropout(0.2) layers.Dense(64, activation='tanh') layers.Dropout(0.1) layers.Dense(6, activation='softmax') optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy']						
Feb 28	Cleansed	1DCNN	15	15	600			layers.Embedding(input_dim=2000, output_dim=**256**, input_length=600) layers.Conv1D(256, 3, padding='same', activation='relu') layers.MaxPooling1D(3, 3, padding='same') layers.Conv1D(32, 3, padding='same', activation='relu') layers.Flatten() layers.Dropout(**0.2**) layers.BatchNormalization() layers.Dense(256, activation='relu') layers.Dropout(0.2) layers.Dense(6, activation='softmax') optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accuracy']	128	31	-	-	-	51.25 % (+/- 5.45%)

NLP Methods Summarised:

Table 5.1.2-1 Performances of My NLP Models Using Hold-out Cross Validation

Date	Random seeds				Vectorizer	Classifier	Cross validation method	Make pipeline ?	Accuracy	Precision (weighted avg)	Recall (weighted avg)	F1 (weighted avg)
	random. seed	np.random. seed	tf.random. set_seed	classifier random_ state								
Dec 09	15	15	15	0	TF-IDF	Support Vector Classifier	Hold-out	Yes	0.86	0.87	0.86	0.85
Dec 09	15	15	15	0	Count Vectorizer	Support Vector Classifier	Hold-out	Yes	0.86	0.87	0.86	0.85
Dec 09	15	15	15	0	Bigram	Support Vector Classifier	Hold-out	Yes	0.68	0.70	0.68	0.68
Jan 31	15	15	15	-	TF-IDF	Multinomial Naive Bayes	Hold-out	Yes	0.78	0.80	0.78	0.76
Jan 31	15	15	15	-	Count Vectorizer	Multinomial Naive Bayes	Hold-out	Yes	0.78	0.79	0.78	0.78
Jan 31	15	15	15	-	Bigram	Multinomial Naive Bayes	Hold-out	Yes	0.74	0.75	0.74	0.72
Feb 21	15	15	15	15	TF-IDF	Random Forest	Hold-out	Yes	0.73	0.74	0.72	0.71
Feb 21	15	15	15	15	Count Vectorizer	Random Forest	Hold-out	Yes	0.69	0.68	0.69	0.68
Feb 21	15	15	15	15	Bigram	Random Forest	Hold-out	Yes	0.54	0.64	0.54	0.56
Feb 21	15	15	15	-	TF-IDF	K-Nearest Neighbors	Hold-out	Yes	0.82	0.86	0.82	0.82
Feb 21	15	15	15	-	Count Vectorizer	K-Nearest Neighbors	Hold-out	Yes	0.55	0.60	0.55	0.56
Feb 21	15	15	15	-	Bigram	K-Nearest Neighbors	Hold-out	Yes	0.26	0.54	0.26	0.18

Table 5.1.2-2 Performances of My NLP Models Using K-Fold Cross Validation

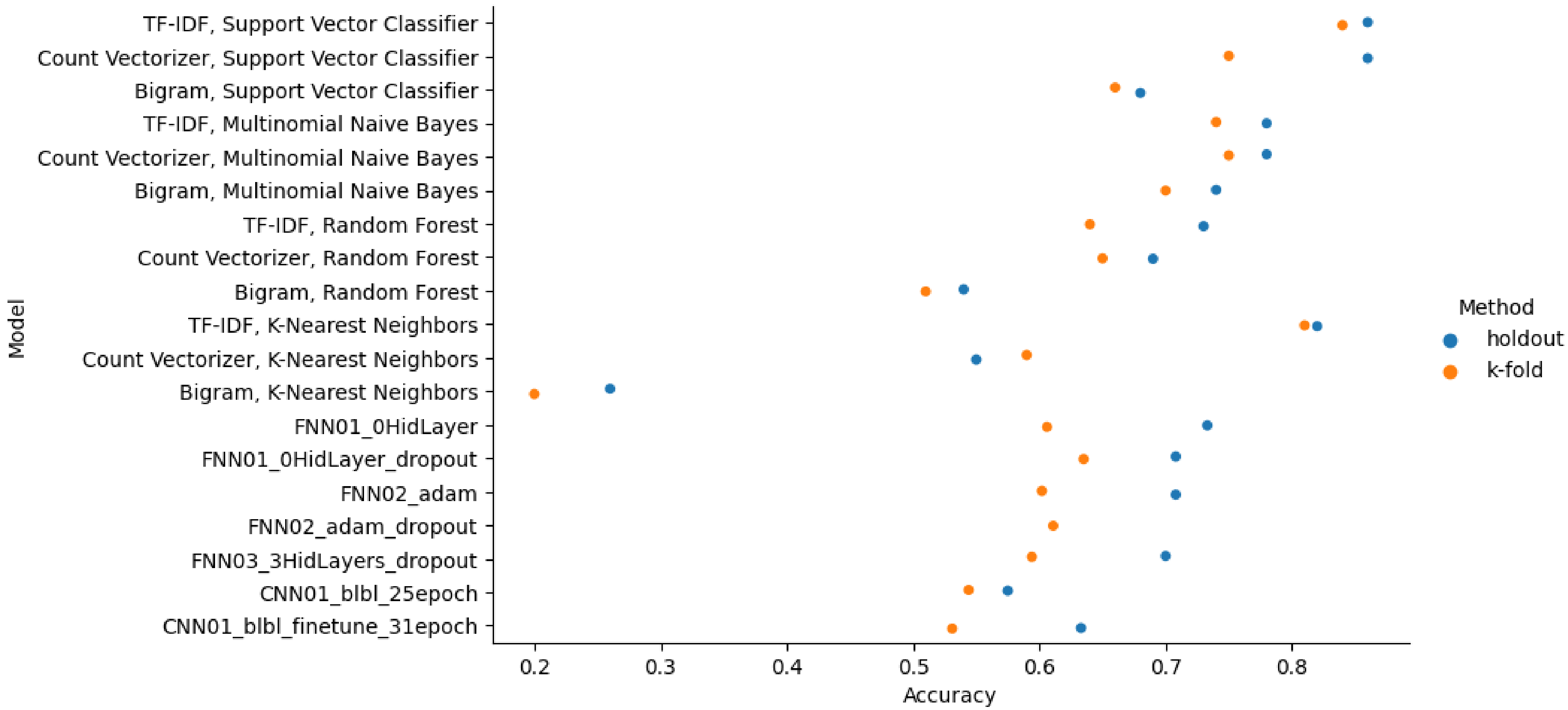
Date	Random seeds				Vectorizer	Classifier	Cross validation method	Make pipeline?	Error?	Avg accuracy with 95% confidence interval
	random. seed	np.random. seed	tf.random. set_seed	classifier random_ state						
Feb 28	15	15	15	0	TF-IDF	Support Vector Classifier	5-fold	yes	-	0.84 (+/- 0.05)
Feb 28	15	15	15	0	Count Vectorizer	Support Vector Classifier	5-fold	yes	warning	0.75 (+/- 0.09)
Feb 28	15	15	15	0	Bigram	Support Vector Classifier	5-fold	yes	warning	0.66 (+/- 0.05)
Feb 28	15	15	15	-	TF-IDF	Multinomial Naive Bayes	5-fold	yes	-	0.74 (+/- 0.08)
Feb 28	15	15	15	-	Count Vectorizer	Multinomial Naive Bayes	5-fold	yes	-	0.75 (+/- 0.08)
Feb 28	15	15	15	-	Bigram	Multinomial Naive Bayes	5-fold	yes	-	0.70 (+/- 0.03)
Feb 28	15	15	15	15	TF-IDF	Random Forest	5-fold	yes	-	0.64 (+/- 0.02)
Feb 28	15	15	15	15	Count Vectorizer	Random Forest	5-fold	yes	-	0.65 (+/- 0.04)
Feb 28	15	15	15	15	Bigram	Random Forest	5-fold	yes	-	0.51 (+/- 0.08)
Feb 28	15	15	15	-	TF-IDF	K-Nearest Neighbors	5-fold	yes	-	0.81 (+/- 0.07)
Feb 28	15	15	15	-	Count Vectorizer	K-Nearest Neighbors	5-fold	yes	-	0.59 (+/- 0.14)
Feb 28	15	15	15	-	Bigram	K-Nearest Neighbors	5-fold	yes	-	0.20 (+/- 0.03)

Table 5.1.2-3 Performances of My NLP Models Using Hold-Out Cross Validation vs. K-Fold Cross Validation

Vectorizer	Classifier	Avg Accuracy using Hold-out	Avg Accuracy using 5-Fold
TF-IDF	Support Vector Classifier	0.86	0.84 (+/- 0.05)
Count Vectorizer	Support Vector Classifier	0.86	0.75 (+/- 0.09)
Bigram	Support Vector Classifier	0.68	0.66 (+/- 0.05)
TF-IDF	Multinomial Naive Bayes	0.78	0.74 (+/- 0.08)
Count Vectorizer	Multinomial Naive Bayes	0.78	0.75 (+/- 0.08)
Bigram	Multinomial Naive Bayes	0.74	0.70 (+/- 0.03)
TF-IDF	Random Forest	0.73	0.64 (+/- 0.02)
Count Vectorizer	Random Forest	0.69	0.65 (+/- 0.04)
Bigram	Random Forest	0.54	0.51 (+/- 0.08)
TF-IDF	K-Nearest Neighbors	0.82	0.81 (+/- 0.07)
Count Vectorizer	K-Nearest Neighbors	0.55	0.59 (+/- 0.14)
Bigram	K-Nearest Neighbors	0.26	0.20 (+/- 0.03)

Accuracy Comparison for Each Good Model Using Hold-Out vs. K-Fold

model	Avg Accuracy using Hold-out	Avg Accuracy using K-Fold
TF-IDF, Support Vector Classifier	0.86	0.84 (+/- 0.05)
Count Vectorizer, Support Vector Classifier	0.86	0.75 (+/- 0.09)
Bigram, Support Vector Classifier	0.68	0.66 (+/- 0.05)
TF-IDF, Multinomial Naive Bayes	0.78	0.74 (+/- 0.08)
Count Vectorizer, Multinomial Naive Bayes	0.78	0.75 (+/- 0.08)
Bigram, Multinomial Naive Bayes	0.74	0.70 (+/- 0.03)
TF-IDF, Random Forest	0.73	0.64 (+/- 0.02)
Count Vectorizer, Random Forest	0.69	0.65 (+/- 0.04)
Bigram, Random Forest	0.54	0.51 (+/- 0.08)
TF-IDF, K-Nearest Neighbors	0.82	0.81 (+/- 0.07)
Count Vectorizer, K-Nearest Neighbors	0.55	0.59 (+/- 0.14)
Bigram, K-Nearest Neighbors	0.26	0.20 (+/- 0.03)
FNN01_0HidLayer	0.733	0.606 (+/- 4.79%)
FNN01_0HidLayer_dropout	0.708	0.635 (+/- 6.67%)
FNN02_adam	0.708	0.602 (+/- 5.70%)
FNN02_adam_dropout	-	0.610 (+/- 6.85%)
FNN03_3HidLayers_dropout	0.700	0.594 (+/- 6.86%)
CNN01_blbl_25epoch	0.575	0.544 (+/- 3.54%)
CNN01_blbl_finetune_31epoch	0.633	0.531 (+/- 4.39%)



(Fig 5.2-1 Accuracy Comparison for Each Good Model Using Hold-Out vs. K-Fold)