AIAA AVIATION Forum
17-21 June 2019, Dallas, Texas
AIAA Aviation 2019 Forum

10.2514/6.2019-2938

Mining and Classifying Aviation Accident Reports

Prabhakar Srinivasan[1] , Venkataramana Nagarajan[2], Sankaran Mahadevan[3]
Vanderbilt University, Nashville, TN 37235, USA

In this paper, we investigate how text data from post-accident analysis reports from the National Transportation Safety Board (NTSB) can be used to develop a knowledge base for real-time safety prognosis. The focus here is on learning with text data that describes sequences of events. Classification models are developed for Class A passenger air carriers, that take either an observed sequence of events or the corresponding raw text narrative as input and make predictions regarding whether an accident or an incident is the likely outcome, whether the aircraft would be damaged or not, and whether any fatalities are likely or not. The classification models are developed using Word Embedding and the Long Short-term Memory (LSTM) algorithm. The proposed methodology is implemented in two steps: (i) transform the NTSB data extracts into labeled datasets for building supervised machine learning models; and (ii) develop deep learning-based classification models for the prediction of adverse events related to accidents, aircraft damage or fatalities.

I. Introduction

The demand for air transportation is continuing to increase around the world, and is expected to nearly double over the next 20 years [1], leading to increased airspace congestion and workload on Air Traffic Control (ATC) personnel. Therefore, it is crucial to develop advanced methodologies that increase air transportation safety. An important step in this regard is to learn valuable lessons from previous accidents and incidents in order to develop appropriate corrective and mitigative measures. The National Transportation Safety Board (NTSB) creates accident investigation reports, which contain raw text narratives as well as the event sequences that pertain to any historical accident or incident in civil aviation. By mining the historical data in NTSB reports, we aim to detect patterns in either the event sequences or in the raw text narratives that can be used to predict a potentially adverse situation.

In this paper, the unstructured text data in NTSB reports (either raw text narratives or event sequences) are transformed into structured data using the Word Embedding technique [2, 3]. This technique is used to convert words in a text into a vector representation. The resulting vectors are used to develop deep learning models that make predictions on whether a sequence of events would lead to accidents, aircraft damage, or fatality.

Federal Aviation Administration (FAA) inspectors routinely conduct inspections and file reports that are archived in a repository called Program Tracking and Reporting Subsystem (PTRS). Jeske and Liu [4] demonstrated a technique to mine the textual PTRS data in order to predict safety violations. In addition, Aviation Safety Reporting System (ASRS) and NTSB are data sources consisting of categorical attributes and descriptive textual narratives of aviation incidents and accidents. A recent work by Zhang and Mahadevan [5] analyzed the event synopses in ASRS using natural language processing (NLP) techniques like bag-of-words analysis to develop Support Vector Machine [6] and Deep Neural Network [7] models to estimate the risk of event consequence given the operational conditions. Similar to the event synopses available in ASRS, the NTSB reports have narratives which are descriptive raw text data. Prior research by Bazargan et al. [8] analyzed NTSB text data using Principal Component Analysis (PCA) [9] and K-Means clustering [10], in order to identify the patterns in general aviation accidents. Reddy et al. [11] also analyzed NTSB data using the random forest technique [12], with a focus on categorical variables, such as, aircraft type, engine, injuries, fatalities, weather conditions, phase of flight, etc. in order to identify the important factors for aviation

[1] Research Engineer, Department of Civil and Environmental Engineering, Vanderbilt University, Nashville, TN.
[2] Graduate Student, Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN.
[3] Professor, Department of Civil and Environmental Engineering, Vanderbilt University, Nashville, TN; AIAA Fellow.

accident risk. Both these approaches have their limitations since they ignore the temporal context and ordering of events in an event sequence that contribute towards an adverse scenario.

A key departure from prior work, and the main motivation for this paper, is to build the capability to analyze event sequences and raw text narratives in NTSB reports, using word embedding and Sequential Deep Learning models. Adapting such innovative techniques that have worked successfully in other domains, in order to analyze NTSB reports and make predictions for adverse scenarios in aviation, is explored in this paper.

Section II describes the methodology developed to access and manage the NTSB data extract and transform it into a format that can be queried easily using a database. Section III describes the natural language processing (NLP) and deep learning (DL) techniques used to build binary classification models capable of predicting accident vs incident, aircraft damage vs no damage, and fatality vs no fatality scenarios, and provides numerical results regarding model performance. Section IV provides concluding remarks.

II. Data Processing

The NTSB website contains reports of all aviation accidents that occurred from 1982 till the present. The data is available as reports and also as data extracts in Microsoft Access database format [13]. NTSB data extracts are laid out in a relational database format[14]. In this study, the data extract is converted into a MySQL data format using database utilities. This makes the data easy to query and analyze. The main database tables of interest are *events, narratives, seq_of_events* (sequence of events), and aircraft. The relationship between the various tables is made available by NTSB [24]. The primary key *ev_id* (event ID) in the events table links to the other tables as well. This key uniquely identifies each accident or incident. Using this key, we can join multiple tables and access the raw text narratives for an accident or incident from the narratives table. Similarly, we can join the events table with the *seq_of_events* table using the *ev_id* key and enumerate all the events in the order of occurrence as reported in the investigation. Since the events in the *seq_of_events* table are standardized using error codes that NTSB has defined, we have a standard dictionary of possible events for describing a scenario.

The identifier *far_part* in the aircraft table with a value of 121 corresponds to commercial civil aviation restricted to passenger flights. The attribute *narr_accf* available in the narratives table provides detailed information in raw text format. The raw textual input is sometimes preferred to the event sequence since it is more descriptive and informative. In this paper, we compare both raw text and event sequences as suitable inputs to the machine learning process.

A set of custom Extract-Transform-Load (ETL) [15][16] data engineering operations in SQL are used to convert the relational data into a single comma-separated file (CSV) format. The *ev_type* attribute serves as an outcome variable, making the CSV dataset ready to serve as input to a supervised machine learning model to predict Accidents vs Incidents. Figure 1 illustrates the ETL data-processing steps.

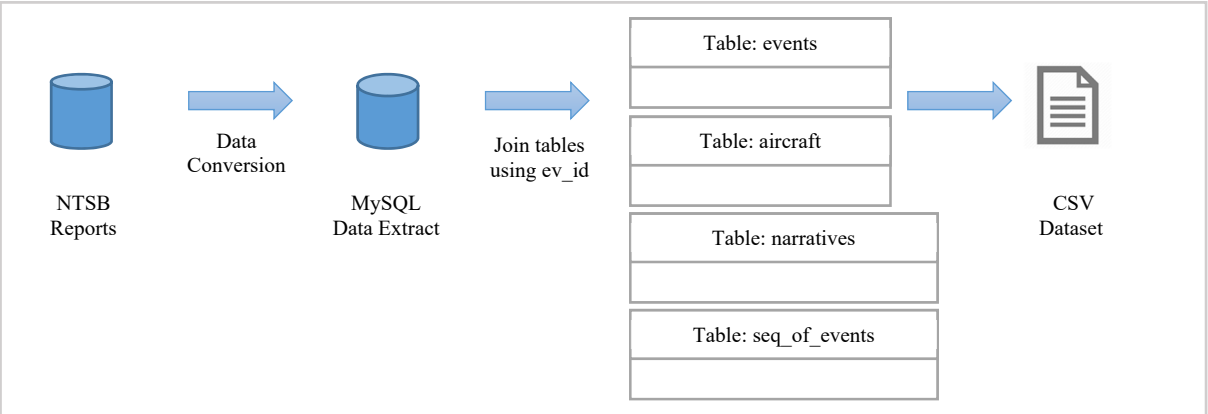

Figure 1. ETL data pipeline to create a CSV dataset

Table 1 illustrates the narratives corresponding to two sample NTSB event IDs 20001208X06203 and 20001208X07738, and Table 2 illustrates the corresponding event sequences.

Table 1. Example of supervised machine learning data in CSV format

ev_id	ev_type	damage	injury level	narr_accf
20001208X06203	ACC	SUBS	FATL	During the initial part of its takeoff roll, the airplane experienced an engine failure. Uncontained engine debris from the front compressor front hub (fan hub) of the #1 (left) engine penetrated the left aft fuselage. Two passengers were killed and two others were seriously injured. The takeoff was rejected, and the airplane was stopped on the runway. The fan hub had fractured through a tierod hole and blade slot. Some form of drill breakage or drill breakdown, combined with localized loss of coolant and chip packing, had occurred during the drilling process, creating an altered microstructure and ladder cracking in the fan hub. Drilling damage extended much deeper into hole sidewall material than previously anticipated by p & w. Fatigue cracks initiated from the ladder cracking in the tierod hole and began propagating almost immediately after the hub was put into service in 1990. The crack was large enough to have been detectable during the last fluorescent penetrant inspection at delta. Delta's nondetection of the crack was caused either by a failure of the cleaning and fluorescent penetrant inspection processing, a failure of the inspector to detect the crack, or some combination of these factors.
20001208X07738	ACC	NONE	NONE	Beech 99 (b99) departed on vfr flight (flt) & began climb to 10,500', heading 175 d eg. At boundary of class b airspace, atc controller (tracon) terminated radar service as b99 was at 7,000'; 40 sec later, pilots (plts) of boeing 737 (b737) contacted traco n, on cresso 3 arrival (star), & were cleared to descend to 10,000' on converging ve ctor of 020 deg; 50 sec later, tracon advised b737 of traffic (b99), 3 mi ahead, oppo site direction & 1,700' lower. Tracon did not mention traffic was climbing, nor issu e safety alert; 18 sec later, tracon told b737 plts they could 'climb as you wish.' 1st officer (fo, plt flying tcas equipped b737) responded to a resolution advisory by us ing autopilot & neglected to press 'level change,' which delayed avoidance maneuv er. B737 plts saw b99 at close range; fo maneuvered abruptly to avert collision. B7 37 flight attendant was injured during maneuver. B99 plt saw b737 behind window post, too late for evasive action. Aircraft passed with estimated 200' vertical & no l ateral separation (sepn). Tcas event analysis concluded that if b737 pilots had resp onded to tcas advisories, within system design parameters, there would have been 7 00' vertical sepn without abrupt maneuvering. Star was in heavily traveled area, oft en used by plts of small vfr aircraft.

Table 2. Event Sequences created from Subj_code meaning and Modifier_Code meaning

ev_id	ev_type	damage	injury level	evt_seq
20001208X06203	ACC	SUBS	FATL	1 ENGINE (NO MODIFIER SPECIFIED), COMPRESSOR ASSEMBLY ROTOR DISC_FATIGUE, MAINTENANCE INSPECTION_INADEQUATE, COMPRESSOR ASSEMBLY ROTOR DISC FRACTURED, COMPRESSOR ASSEMBLY ROTOR DISC_SEPARATION, MISCELLANEOUS ENGINE UNCONTAINED FAILURE (NO MODIFIER SPECIFIED), FUSELAGE CABIN_FOREIGN OBJECT DAMAGE
20001208X07738	ACC	NONE	NONE	APPROACH/DEPARTURE CONTROL SERVICE INADEQUATE, RADAR SEPARATION NOT MAINTAINED, TRAFFIC ADVISORY INADEQUATE, EVASIVE MANEUVER DELAYED, SUPERVISION INADEQUATE, VISUAL LOOKOUT INADEQUATE, EVASIVE MANEUVER ABRUPT

The result of the aforementioned data processing steps yields three datasets in CSV format, one for each of the binary classification models to be developed in this paper: accident vs incident, damage vs no damage, and fatality vs no fatality. As of January 2019, 1,778 reports that have both event sequence information and raw text narrative (related to Class A commercial airlines with an aircraft code of 121) were used in this study. For each adverse scenario, the number of reports for different categories are shown in Table 3. Note that the reports related to damage and fatality are actually divided into more than two categories. These multiple categories are grouped into two in each case, as shown in Table 3, in order to facilitate the development of binary classification models.

Table 3. Class distribution for various adverse scenarios

	Labels				Total
Accident vs Incident	INCIDENT		ACCIDENT		
	940		838		1778
	Labels				Total
Damage vs No-Damage	NONE	MINOR	SUBSTANTIAL	DESTROYED	
	721	607	380	68	1776
	No-Damage	Damaged			
	721	1055			1776
	Labels				Total
Fatality vs No-fatality	NONE	MINOR	SERIOUS	FATAL	
	1357	331	5	80	1773
	No-fatality			Fatality	
	1693			80	1773

Table 4. Class distribution for training, test and validation data for Accident vs Incident scenario

	Accident vs Incident		Total
Training Data	Incident(0)	Accident(1)	
	658	586	1244
Test Data	Incident(0)	Accident(1)	
	282	252	534
Training Data	Incident(0)	Accident(1)	
	460	410	870
Validation Data	Incident(0)	Accident(1)	
	198	176	374

A test dataset is created with 30% of the 1,778 samples using stratified sampling, yielding 534 samples. The remaining 70% training data has 1244 samples, which consists of 658 incidents and 586 accidents. The training dataset is further split during 5-fold cross-validation at a 70%-30% ratio using stratified sampling. Table 4 illustrates how the training, validation and test data are split for the Accident vs Incident scenario. Similarly, the training, validation and test data are also created for the damage vs no-damage and fatality vs no-fatality models.

III. Model Development

A. Overview of proposed approach

In this work, classification models are developed for three types of outcomes: accident vs incident, damage vs no damage, and fatality vs no fatality, based on two different input types that contain sequential data: raw text narratives (Table 1) converted into word-embedding format, and the corresponding coded event sequences (Table 2) converted into word-embedding format, derived from the NTSB data extracts. In total, six classification models will be built for the combination of two input data formats and three adverse scenarios. The algorithm that is being investigated is a

type of Recurrent Neural Network (RNN) known as Long Short-term Memory (LSTM). In this paper, we train the LSTM on the supervised machine learning datasets and the resulting models are used for doing prognosis of the above adverse outcome events.

B. Narrative raw text data pre-processing: Stop-words, Stemming, and Word-Embedding

The raw input textual data is pre-processed to filter out the stop-words, using the standard dictionary of stop-words available in the Python Natural Language Toolkit (NLTK). This eliminates grammatical parts-of-speech like articles, prepositions etc., which are found commonly in the narratives. In addition, stemming technique is applied to map multiple tenses for a verb to the canonical form in present tense and to convert the singular and plural forms of a word into a single canonical root word element, using the Snowball Stemmer [17] available in the NLTK package.

The next step is to convert word tokens into a vector representation that are compatible with the input data format of LSTM. The Keras tokenizer is utilized here, which preserves the order of the tokens as appearing in the NTSB narratives or event sequences, by replacing the tokens by an integer representation of the token in an ordered dictionary. A dictionary is created using all of the combined vocabulary of the training, validation and test data. The tokens are converted in a vector representation using this dictionary.

Word embedding is a feature representation technique in NLP that encodes words as real-valued vectors in a high dimensional space, where words with similar meaning are close to each other in the vector space. For example, apple and banana would appear close to each other in the vector space after word embedding of documents consisting of these two fruit names. In the context of NTSB reports, two events “radar separation not maintained” and “traffic advisory inadequate” might be close to each other in a vector space if these two events figured prominently in accidents.

C. LSTM Model

The DL algorithm used for learning the prognosis model using the word-embedding format of either the raw text input data or the event sequences as input data is Long Short-term Memory (LSTM) [18, 19], which is a specialized Recurrent Neural Network (RNN) architecture. The LSTM has proven to be efficient at making predictions on ordered sequences of data like time-series data, natural language data, handwriting or music. This makes it an ideal algorithm to use for doing prognosis on event sequences or raw text narratives. While LSTM is a general approach, it needs to be optimized and adapted to the particular application in terms of the choice of number of layers and their ordering, number of cells or units within each layer, activation functions and learning rate, and the number of epochs of learning (i.e., the number of iterations of backpropagation). This tuning is done by trial and error, with the objective of minimizing the log-loss metric for the validation data set (explained below). For the current analysis, the optimum solution consists of an LSTM layer with 64 cells and a Dense layer with 256 neurons and finally a single neuron as the output layer with sigmoid activation function. This configuration was chosen after evaluating a few other configurations and by selecting the one that provides the best accuracy and loss metrics. The L1 loss (log-loss or binary cross-entropy) algorithm is used to train the LSTM model [20].

Overfitting is prevented using a generic Dropout [21, 22] and early-stopping criterion [23]. The term "dropout" refers to temporarily removing a unit from the neural network, along with all its incoming and outgoing connections. The choice of which units to drop is random. In this work, a dropout value of 0.2 proved to be an appropriate value since it gave the best F1-score metric; that is, 20% of neurons and their weights are randomly selected and removed during each epoch of training. The implementation uses Keras version 2.2.2 with Tensorflow 1.5.1 [24] as the backend on a computer with Ubuntu 16.04 operating system.

In the NTSB data, the non-uniform distribution of the binary outcome variables for accident vs incident, damage vs no damage, and fatality vs no fatality, poses a class imbalance problem since the classification algorithms require a uniform distribution of the outcome variable’s values during the training stage. This is a common problem and Buda et al [25] have analyzed various cases of class imbalance with regards to neural networks in detail. Several approaches such as up-sampling and use of class weights are available to overcome this problem; the latter option available in Keras [26] is used here. To ensure that the log-loss is not biased by the majority class, the class weights or class ratios are computed. By using the class weights during the training stage, we can add more penalty to the log-loss of the minority class (accident) and less penalty to the log-loss of the majority class (incident). The class weights can be used

to adjust the stochastic gradient descent such that observations are not picked randomly, assuming uniform class distribution, but rather in the ratio of the outcome variable's class distribution.

D. Model Evaluation

A comparison of the output of the binary classification model and the ground truth results in four possible outputs, namely true positive (TP), false positive (FP), true negative (TN), and false negative (FN). Based on these four quantities, many common model evaluation metrics like Sensitivity, Specificity, Precision, Accuracy and F1-score can be evaluated. Table 3 provides the list of metrics and their definitions.

Table 5. Model evaluation metrics

Metric	Definition
Sensitivity	TP/(TP+FN)
Specificity	TN/(TN+FP)
Precision	TP/(TP+FP)
Recall	TP/(TP+FN)
Accuracy	(TP+TN)/(TP+TN+FP+FN)
F1-score	2 * (Precision * Recall)/ (Precision + Recall)

Table 6. Summary of evaluation metrics for predictions on test data for all six models

	Accident vs Incident		Damage (Y/N)		Fatality (Y/N)	
	Event Seq.	Narratives	Event Seq.	Narratives	Event Seq.	Narratives
TP	182	196	257	288	13	4
TN	229	228	151	150	470	497
FP	53	53	65	66	38	11
FN	70	55	60	29	11	20
TPR (Recall or Sensitivity)	0.7222	0.7809	0.8107	0.9085	0.5417	0.1667
TNR (Specificity)	0.8121	0.8114	0.6991	0.6944	0.9252	0.9783
PPV (Precision)	0.7745	0.7871	0.7981	0.8136	0.2549	0.2667
Accuracy	0.7697	0.7970	0.7655	0.8218	0.9079	0.9417
F1-score	0.7474	0.7840	0.8044	0.8584	0.3467	0.2051

The metrics defined in Table 5 are used for evaluating the predictions of the 5-fold cross-validation, for each of the six classification models developed. The summarized metrics for the predictions on the test data for all six models are reported in Table 6. We observe from the F1-scores that the LSTM does indeed provide a good prognosis for various adverse scenarios. The models which are built using the narratives (which have richer information content) tend to outperform the models built using the event sequences in most cases. Perhaps, if the event sequences capture more of the details then the discrepancy could be reduced.

IV. Conclusion

This paper explored the mining of NTSB reports regarding adverse events in civil aviation, by (i) converting unstructured raw-text narratives and event sequence data into a structured format using word embedding, and (ii) building LSTM models for binary classification of accident vs. incident, damage vs. no damage, and fatality vs. no fatality. The suite of resulting models can also support a prototype query capability that allows the ATC operator to input either complete or partial event sequences or natural language text and estimate the probability of adverse

outcomes like accidents, damage or fatality. This capability could be used to perform what-if scenario analyses and identify possible remedial actions.

The models based on raw text narratives can be further improved by standardization of abbreviations used in the raw text narratives, and additional noise reduction and dimension reduction techniques. The models based on event sequences can be further improved by additional manual encoding of more of the rich content in the raw text narratives in the form of more detailed event sequences; in some records, there is currently only a single event listed in the event sequence data.

Acknowledgment

The research reported in this paper was supported by funds from the NASA University Leadership Initiative program (Grant No. NNX17AJ86A, Project Technical Monitor: Dr. Anupa Bajwa) through subcontract to Arizona State University (Principal Investigator: Dr. Yongming Liu). The support is gratefully acknowledged.

References

1. IATA, 2017. 2036 Forecast Reveals Air Passengers Will Nearly Double to 7.8 Billion, URL: https://www.iata.org/pressroom/pr/Pages/2017-10-24-01.aspx
2. Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. The Journal of Machine Learning Research, 3:1137–1155, 2003
3. T. Mikolov et al., 2013. Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems.
4. D. R Jeske & R. Y Liu (2007), 2012. Mining and Tracking Massive Text Data: Classification, Construction of Tracking Statistics, and Inference Under Misclassification, Technometrics, 49 (2) 116– 128
5. X. Zhang and S. Mahadevan. 2019. Ensemble machine learning models for aviation incident risk prediction. Decision Support Systems volume 116, Jan 2019, Pages 48-63
6. C. Cortes and V. Vapnik. 1995. Support vector networks. Machine Learning, 20:1–25, 1995
7. Y. LeCun, Y. Bengio, G. Hinton Deep learning. Nature, 521 (7553) (2015), pp. 436-444
8. M. Bazargan, M. Johnson, A. Vijayanarayanan et al. 2013. An Evaluation of AIRES and STATISTICA Text Mining Tools as Applied to General Aviation Accidents. DOT/FAA/TC-TN13/7
9. H. Hotelling. 1933. Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology 24:417–441, 498–520
10. H. Steinhaus. 1956. Sur la division des corp materiels en parties. Bull. Acad. Polon. Sci. 1, 801
11. N. Reddy and M. Padma. 2017. NTSB Aviation Accidents Analysis Using R, International Journal of Emerging Trends Technology in Computer Science (IJETTCS) Vol. 6, Issue 4.
12. L. Breiman. 2001. Random forests. Machine Learning, 45 (1) (2001), pp. 5-32
13. NTSB data extract [online] https://app.ntsb.gov/avdata/Access/
14. NTSB entity relationship diagram [online] https://app.ntsb.gov/avdata/eadmspub.pdf
15. P. Vassiliadis, A. Simitsis, and E. Baikousi. 2009. A Taxonomy of ETL Activities. Proceedings of the ACM Twelfth International Workshop on Data Warehousing and OLAP, New York, NY, USA, 2009, pp. 25-32
16. P. Vassiliadis, A. Simitsis, P. Georgantas, M. Terrovitis, and S. Skiadopoulos. 2005. A generic and customizable framework for the design of ETL scenarios. Information Systems, vol. 30, no. 7, pp. 492-525, Nov. 2005
17. M Porter. 2001. Snowball: A language for stemming algorithms. URL: http://snowball.tartarus.org/texts/
18. C. Olah. 2015. Understanding LSTM Networks. Retrieved from http://colah. github.io/posts/2015-08-Understanding-LSTMs/
19. K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and Jürgen Schmidhuber. 2017. LSTM: A Search Space Odyssey. IEEE Transactions on Neural Networks and Learning Systems, vol. 28, no. 10, October 2017
20. C.M. Bishop. 2006. Pattern Recognition and Machine Learning. Springer, p. 209.
21. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. JMLR 15(1), 1929–1958 (2014)
22. Y. Gal, Z. Ghahramani. 2016. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. NIPS ’16 Proceedings of the 30th International Conference on Neural Information Processing Systems, pp. 1027-1035
23. L. Prechelt. 1998. Automatic early stopping using cross validation: quantifying the criteria. Neural Networks. 11(4), 761–767 (1998)

24. M. Abadi, A. Agarwal et al. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. URL: http://tensorflow.org/
25. M. Buda, A. Maki, M. A. Mazurowski. 2017. A Systematic Study of the Class Imbalance Problem in Convolutional Neural Networks. arXiv preprint arXiv:1710.05381
26. F. Chollet et al. Keras. 2015. URL: https://keras.io