SportSett:Basketball - A robust and maintainable dataset for Natural Language Generation

Craig Thomson, Ehud Reiter, and Somayajulu Sripada

Department of Computing Science, University of Aberdeen:
{c.thomson, e.reiter, yaji.sripada}@abdn.ac.uk

Abstract

Data2Text Natural Language Generation is a complex and varied task. We investigate the data requirements for the difficult real-world problem of generating statistic-focused summaries of basketball games. This has recently been tackled using the Rotowire and Rotowire-FG datasets of paired data and text. It can, however, be difficult to filter, query, and maintain such large volumes of data. In this resource paper, we introduce the SportSett:Basketball database[1]. This easy-to-use resource allows for simple scripts to be written which generate data in suitable formats for a variety of systems. Building upon the existing data, we provide more attributes, across multiple dimensions, increasing the overlap of content between data and text. We also highlight and resolve issues of training, validation and test partition contamination in these previous datasets.

1 Introduction

Natural Language Generation (NLG), particularly at the document planning level, has traditionally been tackled using template (McKeown, 1985) or rules-based solutions (Mann and Thompson, 1988; Reiter and Dale, 2000). Statistical methods have also been explored (Duboue and McKeown, 2003). More recently, it has become popular to frame the NLG problem as one of sequence-to-sequence (Seq2Seq) modelling, where the input of an encoder-decoder architecture is the data, and the target output is human-authored text (Sutskever et al., 2014; Cho et al., 2014; Bahdanau et al., 2014; Lebret et al., 2016). A discussion of most techniques can be found in the most recent survey of the NLG field (Gatt and Krahmer, 2018).

Data2Text NLG applications that use classical rules-based methods are currently not comparable with state-of-the-art Seq2Seq applications. For Seq2Seq applications, the input data is either very shallow, or the output texts exhibit a high degree of factual inaccuracy. The task here is very different to others such as chat bots (Adiwardana et al., 2020), where the focus is on appearing human, rather than conveying concise yet relevant information.

A lot of research has been done with datasets created for the WebNLG (Gardent et al., 2017) and E2E (Dušek et al., 2018) challenges. There is also the more recent ToTTo dataset (Parikh et al., 2020). Such datasets can provide interesting sentence-level insights, although the data structures are quite simple (a single table or simple schema) and the human-authored texts are short (usually one or two sentences). This makes them unsuitable for evaluating Data2Text systems which generate summaries based on more complex data analytics. This is not necessarily because the systems are incapable of doing so, but because the data is not available as input. In addition, rules-based systems for problems based on these datasets are neither difficult or time-consuming to implement, meaning it is harder to investigate their limitations under these conditions.

The Rotowire dataset of basketball game summaries (Wiseman et al., 2017), along with the expanded (in terms of game count) Rotowire-FG (Wang, 2019), have become popular resources for investigating the generation of paragraph-sized insightful texts. Several different Seq2Seq models have been proposed, evaluated, and compared using them (Wiseman et al., 2017; Puduppully et al., 2019a,b; Wang, 2019; Gong et al., 2019; Rebuffel et al., 2020). The datasets consist of basketball box scores (see Table 1) and human-authored game summaries (see Figure 2). The example sentence in Figure 1 highlights the level of complexity in these summaries. It includes a set of average statistics for a player over multiple games, as well as the claim that this means the player 'stayed dominant'. This

[1] https://github.com/nlgcat/sport_sett_basketball

is just one sentence of many in the full summary. We found Rotowire to be unsuitable for evaluating this in its current format. However, the domain itself is suitably complex and Rotowire provides a foundation upon which we can build.

Figure 1: Example sentence from Rotowire-FG. Full text in Figure 2

> He's continued to stay dominant over his last four games, averaging 27 points, 11 rebounds and 2 blocks over that stretch.

As part of a PhD project, we are creating a hybrid document planning solution that combines rules-based and machine learning methods. Our hybrid system will use known relationships and simple rules to manipulate a predicate-argument schema based on that of Construction-Integration theory (Kintsch, 1998). It will then learn to perform parts of the NLG process that are overly complex or time consuming to manually define. Like any Data2Text system, we require sufficient data, both in terms of quantity and complexity, to provide a difficult and realistic problem. Our NLG system is not discussed further here, it was mentioned in order to illustrate the problems we encountered on the data side when implementing it. The data issues are the focus of this paper.

In this resource paper, we identify issues with the structure and quality of the existing Rotowire based datasets. We then introduce an alternative, the SportSett:Basketball database. In addition to improving the quality and quantity of data, SportSett:Basketball stores game statistics in a hierarchy which can be queried in multiple dimensions. This allows for a richer entity relationship graph, the exploration of which we hope will enable future research in this challenging area. Serving as a master source, data can be exported in a range of formats, from rich graphs for Rhetorical Structure Theory based systems (Mann and Thompson, 1988), to tables or flat files for machine translation based systems.

2 The SportSett:Basketball database

SportSett:Basketball is a PostgreSQL (The PostgreSQL Global Development Group) relational database (Codd, 1970), with (optional) object-relational mappings (ORM) written in Ruby Sequel (Jeremy Evans). It provides researchers with the ability to query and filter data, in a simple and efficient way. The process of importing data into a normalized relational database also helps to verify the data, clean it, and eliminate redundancy. By writing simple scripts, either in SQL or using the ORM, data can be easily output in the format a researcher requires for their system. We tested this functionality by creating data for a recent OpenNMT based system (Klein et al., 2017; Rebuffel et al., 2020) (see section 3 for details).

There are problems with the structure, quality and partitioning of existing Rotowire based datasets. Our investigation focuses on Rotowire-FG as it contains more games, although the underlying problems are the same in both datasets. Some of these issues are minor, such as the team a player is on being indexed by city rather than name (there are two teams in Los Angeles, the Clippers and the Lakers). Others, like the partition contamination discussed in subsection 2.3 are more serious. The JSON file format also becomes unwieldy as data size and complexity increases, especially when researchers need to perform tasks like checking claims made in generated text relative to input data. For brevity, we do not discuss every minor change we have made here. The repository which will host this data resource will include an in-depth discussion for those who are interested.

The sequential nature of a season is modelled SportSett:Basketball, with each of the 82 regular season games for each team during the 2014 through 2018 seasons. The database does support preseason and playoff games, although the data for them has yet to be imported and verified with the level of scrutiny that regular season games have. Data sources include rotowire.com, basketball-reference.com and wikipedia.com. A UML class diagram of the object-relational mappings can be found in Figure 5 in the appendix.

Other efforts have been made to improve existing datasets. Whilst what we propose here is different to dataset cleaning techniques such as those applied to the E2E dataset (Dušek et al., 2019), we do not view them as mutually exclusive. Such automated techniques could be tried on existing datasets, as well as the new SportSett:Basketball. What we propose is a more fundamental redesign of the data, using traditional data-modelling techniques.

Table 1: Example partial box score for NOP@OKC on February 4th 2015, showing Oklahoma starters. Full box scores show approx 24 players.

Player	MP	FG	FGA	FG%	3P	3PA	3P%	FT	FTA	FT%	REB	AST	STL	BLK	TOV	PTS
Serge Ibaka	41:36	6	9	.667	1	2	.500	0	0	N/A	6	0	0	7	0	13
Russell Westbrook	40:28	18	31	.581	2	6	.333	7	9	.778	6	6	1	1	6	45
Dion Waiters	33:06	6	14	.429	0	2	.000	0	2	.000	6	2	2	1	4	12
Steven Adams	25:04	4	7	.571	0	0	N/A	0	0	N/A	10	4	0	0	2	8
Andre Roberson	11:44	0	0	N/A	0	0	N/A	0	0	2	1	0	0	0	1	0

2.1 Dimensions, Sets, Entities and Attributes

Entities in the database represent people (such as players), real objects (such as stadia), events (seasons, games, periods, plays), as well as conceptual objects (like statistics). When using the object-relational mappings, an entity will normally be represented by an object which maps to a tuple in the relational database. Attributes, such as player names or game dates, are mapped to database columns.

A dimension is any axis along which we can group or filter these entities. Dimensions can be simple or complex. A simple dimension occurs when entities can be filtered independently by one of their attributes, such as all games on a given calendar data, or all players with the surname 'Antetokounmpo'. A complex dimension occurs when entities can be arranged in hierarchical sets, such as a person-in-a-game or a team-in-a-game-period. In such cases, the entity would not make sense without both of its parent sets (a person-in-a-game makes no sense without a game, or a person). This is different to previous work on dimensionality on Rotowire which defined three dimensions of time, along with the rows and columns of the box score (Gong et al., 2019). In order to include all data which could be included in the text, we need to include all dimensions, starting from those comprised on atomic entities.

Within the complex dimensions we have identified for the NBA, entities and events at the same level within the hierarchy do not overlap. Players or teams never play in two games simultaneously, similarly, seasons are disjoint sets of games. This removes the need for a complex model of events (Allen, 1983) and is why we can model entities in such a hierarchy.

2.1.1 Sets of People

The atomic entity within this hierarchy is a single person (usually a player although it could be a coach or official). People are grouped together in teams, with teams then grouped within some form of league structure. In the case of the NBA, the league consists of 2 conferences, each containing 3 divisions, of 5 teams. Each team then has a roster of at most 17 players. These groupings are shown in Table 2.

Table 2: Data coverage, dimensions in Rotowire are also in SportSett. Since SportSett models the atomic entities of Person and Play, the entire grid can be extrapolated.

	Events				
People	**All**	**Season**	**Game**	**Period**	**Play***
League					
Conference					
Division					
Team			◇	□	
Person*			◇	△	△

◇ in Rotowire □ partially in Rotowire
△ added in SportSett
* atomic entity

2.1.2 Sets of Events

We define atomic events as plays. Each play covers a span of time where something happened in the game which caused a statistic to be counted. In most cases, either one or two players will be involved in a play, for example one player may attempt a shot which another blocks. There are, however, cases where a play refers to a whole team or to the officials. Plays are grouped into game periods, with there being 4 periods (barring overtime) in a game. Games are, in turn, grouped within seasons. The event hierarchy for the NBA is shown for the in Table 2.

These sets of events differ from any temporal dimension as whilst they are played out as an ordered list, the actual date or time does not matter, provided the order is preserved. Our database allows for querying by date, but this is an attribute of an event (a simple dimension).

2.2 Missing Attributes

Whilst it is highly impractical to align data with everything a human could possibly have said in a text

Figure 2: Human-authored basketball summary for NOP@OKC on February 4th 2015

The Oklahoma City Thunder (25-24) defeated the New Orleans Pelicans (26-23) 102-91 on Wednesday at the Smoothie King Center in New Orleans. The Thunder shot much better than the Pelicans in this game, going 53 percent from the field and 33 percent from the three-point line, while the Pelicans went just 39 percent from the floor and a meager 28 percent from beyond the arc. While the Thunder were down 57-51 at half, they had a huge second half where they out-scored the Pelicans 51-34, allowing them to steal a victory over the Pelicans. It was a big win, as they will be fighting against the Pelicans to secure one of the last spots in the Western Conference playoffs moving forward. With Kevin Durant still sitting out with a toe injury, Russell Westbrook again took it on himself to do the bulk of the work offensively for the Thunder. Westbrook went 18-for-31 from the field and 2-for-6 from the three-point line to score a game-high of 45 points, while also adding six rebounds and six assists. It was his second 40-point outing in his last four games, a stretch where he's averaging 31 points, 7 rebounds and 7 assists per game. Serge Ibaka had a big game defensively, as he posted seven blocks, to go along with 13 points (6-9 FG, 1-2 3Pt) and six rebounds. Over his last two games, he 's combined for 29 points, 14 rebounds and 10 blocks. Dion Waiters was in the starting lineup again with Durant out. He finished with 12 points (6-14 FG, 0-2 3Pt, 0-2 FT), six rebounds and two steals in 33 minutes. The only other Thunder player to reach double figures was Anthony Morrow, who had 14 points (6-11 FG, 2-4 3Pt) and four rebounds off the bench. The Pelicans got most of their production from Anthony Davis, who posted 23 points (9-21 FG, 5-6 FT) and eight rebounds in 39 minutes. He's continued to stay dominant over his last four games, averaging 27 points, 11 rebounds and 2 blocks over that stretch. Giving Davis the most help was Ryan Anderson, who came off the bench to score 19 points (7-17 FG, 3-8 3Pt, 2-2 FT), to go along with five rebounds and two steals in 28 minutes. He 's been the most reliable player off the bench for the Pelicans this season, so it was good to see him have another positive showing Wednesday. Despite shooting quite poorly, Tyreke Evans came close to a triple-double, finishing with 11 points (5-20 FG, 1-5 3Pt), seven rebounds and seven assists. Quincy Pondexter reached double figures as well and posted 10 points (4-8 FG, 2-5 3Pt) and seven rebounds off the bench. These two teams will play each other again on Friday in Oklahoma.

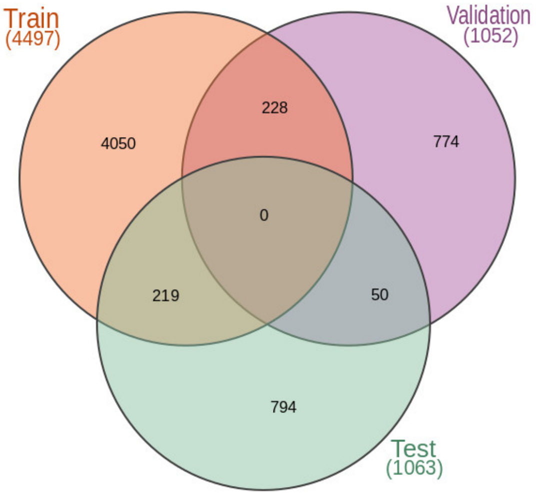
corpus, this does not mean that arbitrarily taking a limited set of data is sufficient. A basic corpus analysis, either manually or with information extraction tools, will show some common phrases and patterns in the text. In the case of Rotowire summaries, the texts frequently mention the day of the week on which a game was played, as well as its location (place and/or stadium). It is also common for a text to state the next opponent for both teams and where those games will be played.

Rotowire does not include the name of the stadium where the game was played. NBA teams do not necessarily play each home game in the same stadium. For NBA International games, a team will play what counts as a home game but in a city outside the U.S. such as London or Paris. A team might also temporarily relocate due to stadium problems or construction. The SportsSett:Basketball database adds attributes for the stadium name and location, as well as more convenient methods of accessing data for previous or subsequent games.

2.3 Partition contamination

It is crucial to ensure that machine learning systems generalise for unseen data. This is usually accomplished by withholding part of the data from the training process, in order to provide for a fair evaluation. Whilst this common train/validate/test partition scheme was adopted in both of the Rotowire based datasets, there are contamination problems. Figure 3 shows the number of instances where multiple human-authored texts are present for the same game data, but are placed in different Rotowire-FG partitions. This could have occurred where summaries written by different sports journalists have been scraped for the same game. Both the surface form of these texts, as well as the opinions they express, could be very different. As a result of this contamination, systems are evaluated (both in validation and testing) on game data which was previously used to condition the encoder. This could lead to over-fitting of the model.

Figure 3: Venn Diagram showing Partition Contamination in Rotowire-FG. Numbers represent the total games in each set. Numbers in parenthesis indicate the total size of the training, validation and test sets. The level of contamination in the original Rotowire was almost identical.



There are two problems with the existing partition scheme which need to be overcome. Firstly, the partition contamination needs to be removed. This is as simple as only including a game record in one partition, with multiple human-authored texts describing the game being allowed only within this same partition. Secondly, we need to limit contamination as much as possible when data must be used to create aggregate summaries between game events.

This second problem is more complex. Given that human-authored game summaries often include statistics aggregated over several games, it makes sense that a model might take data from more than one game as input (Gong et al., 2019). If this additional data from outwith the game is included in a different data partition then it cannot be used in this way.

We suggest that seasons remain disjoint sets when included in training, validation or test partitions. For example, we use 2014, 2015, and 2016 as training data, 2017 as validation data, then 2018 as withheld test data. This limits contamination as much as is practical and also reflects the scenario in which such a system may be deployed. A sports company such as ESPN might provide data and texts from previous seasons and expect in return an NLG system that generates texts as future seasons play out. We would ideally cross-validate, with different partition setups. However, at about 3 weeks for just one partition setup, we found compute time prohibitive.

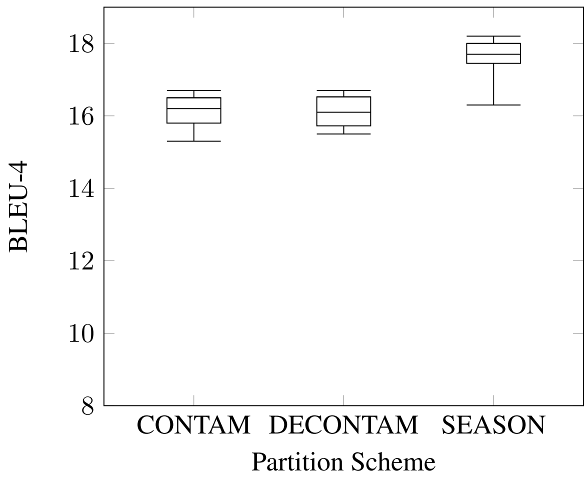
3 Initial Experiments

In order to confirm that our data can be easily used by existing systems, we exported it to the format of one of the more recent Seq2Seq architectures (Rebuffel et al., 2020) then attempted to replicate the results for BLEU, one of the more popular automated metrics (Papineni et al., 2002). BLEU scores do not correlate with human evaluation of text (Reiter, 2018) but may be useful in early system development. Our aim was to ensure that our data, particularly with the new partition scheme, could be used in place of the existing Rotowire datasets with minimal effort. We used the same hyper-paramaters as the original work, except for the batch size which we reduced from 32 to 16 due to hardware constraints, training 10 models with different random seeds for each of the below partition schemes:

- CONTAM: The original partition scheme of Rotowire-FG.
- DECONTAM: Starting with CONTAM, we move entires out of sets until contamination is removed.
- SEASON: Season-based partitions with 2014-16 used for training, 2017 as validation, and 2018 as test.

The data entities and attributes were very similar to the original work, with only slight differences due to minor data changes we have made when creating SportSett:Basketball. Training and evaluating these models took three weeks on a workstation containing a pair of 11GB Nvidia 2080Ti RTX GPUs. For each dataset/seed combination we took the model which performed best against the validation set (checking this every 2000 steps up to 30,000). Weights were then averaged over the previous 4 checkpoints of 500 steps each. We then calculated BLEU using the withheld test set on each final model. Figure 4 shows the BLEU score distribution for each partition scheme.

Figure 4: Box plot showing BLEU scores for each partition scheme.



Using a one-way ANOVA with a post-hoc Tukey test we find no difference between CONTAM AND DECONTAM ($p > 0.5$). We do find the difference between each of these and SEASON to be statistically significant ($p < 0.01$). The results may, however, be sensitive to the choice of partition scheme and therefore no claims are made about the comparative status of these scores. In future, more robust quality measures will be provided.

We also calculated BLEU scores comparing the 2018 test set with a partially shuffled copy of itself (ensuring each game is matched with one other than itself, but the home team is the same). This yielded a score of 8.0 which we use as a baseline, offsetting the y-axis of Figure 4.

It is difficult to determine what effect the contamination of partitions will have had on the results reported in previous work (Wiseman et al., 2017; Puduppully et al., 2019a,b; Wang, 2019; Gong et al., 2019; Rebuffel et al., 2020). Even though the encoder may be conditioned on data which it is then tested with, the target text is different. The system would be learning the style of one author, before being measured against that of a second author. This highlights one of the key failings of n-gram based metrics, there is not only one correct gold-standard text.

The level of factual error we have observed when manually checking a small number of texts ourselves has also been quite high, although further investigation is required in order to ascertain the exact nature and extent of this problem. Metrics based on the overlap of n-grams tell us very little about whether a text has described the relationships between entities across different dimensions correctly.

4 Discussion

The data matters in Data2Text, irrespective of which system architecture is being evaluated. SportSett provides an increased volume of data, as well as an improved structure. The database also allows for researchers to easily query data, from many different dimensions, for output in a variety of formats for different architectures.

Future research will focus on the effect the dimensionality of data outlined in this paper has when generating statistical summaries in the sport domain. This will be investigated both with our hybrid architecture, and Seq2Seq systems. We plan to expand the database to include more sports, since game summaries may differ between them. There are often 75 or more scoring plays in a basketball game, meaning individual plays are usually not mentioned in game summaries unless they occurred on or near the expiration of the game clock. This differs greatly from NFL games (American Football) where even fifteen scoring plays would be considered high. As a result, individual plays feature more heavily in the narrative. We also plan to include human-authored game preview texts which describe upcoming games.

We hope that both the database, along with some of the ideas presented in this paper can be adopted by other researchers looking to solve this complex problem.

Acknowledgments

This work is funded by the Engineering and Physical Sciences Research Council (EPSRC), which funds Craig Thomson under a National Productivity Investment Fund Doctoral Studentship (EP/R512412/1).

We would like to thank our reviewers, as well as the NLG (CLAN) Machine Learning reading groups at the University of Aberdeen for their helpful feedback on this work. We would also like to thank Moray Greig, who was our basketball domain expert.

References

Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. 2020. Towards a human-like open-domain chatbot.

James F. Allen. 1983. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

E. F. Codd. 1970. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387.

Pablo Ariel Duboue and Kathleen R. McKeown. 2003. Statistical acquisition of content selection rules for natural language generation. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 121–128.

Ondřej Dušek, David M. Howcroft, and Verena Rieser. 2019. Semantic noise matters for neural natural language generation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 421–426, Tokyo, Japan. Association for Computational Linguistics.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2018. Findings of the E2E NLG challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 322–328, Tilburg University, The Netherlands. Association for Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *J. Artif. Int. Res.*, 61(1):65–170.

Heng Gong, Xiaocheng Feng, Bing Qin, and Ting Liu. 2019. Table-to-text generation with effective hierarchical encoder on three dimensions (row, column and time). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3143–3152, Hong Kong, China. Association for Computational Linguistics.

Jeremy Evans. Sequel.

Walter Kintsch. 1998. *Comprehension : a paradigm for cognition*, 1st edition. Cambridge University Press.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, Austin, Texas. Association for Computational Linguistics.

W.C. Mann and S.A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281. Cited By 820.

Kathleen R. McKeown. 1985. Discourse strategies for generating natural-language text. *Artif. Intell.*, 27(1):1–41.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. Totto: A controlled table-to-text generation dataset. *arXiv preprint arXiv:2004.14373*.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019a. Data-to-text generation with content selection and planning. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, Honolulu, Hawaii.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019b. Data-to-text generation with entity modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2023–2035, Florence, Italy. Association for Computational Linguistics.

Clément Rebuffel, Laure Soulier, Geoffrey Scoutheeten, and Patrick Gallinari. 2020. A hierarchical model for data-to-text generation. In *Advances in Information Retrieval*, pages 65–80, Cham. Springer International Publishing.

Ehud Reiter. 2018. A structured review of the validity of BLEU. *Computational Linguistics*, 44(3):393–401.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Cambridge University Press, New York, NY, USA.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

The PostgreSQL Global Development Group. PostgreSQL.

Hongmin Wang. 2019. Revisiting challenges in data-to-text generation with fact grounding. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 311–322, Tokyo, Japan. Association for Computational Linguistics.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

A Appendices

A.1 UML Diagram

To highlight the number of tables, attributes and relations, we have included a UML Class diagram for the Ruby ORM (see Figure 5. The data structure may change in the future as it is refined and augmented. Please see the repository for the most up to date version.

The Ruby Sequel library consists of Object-Relational Mappings with database table migrations. Researchers can either use SQL directly, which should be very fast, or the ORM, which will be slower but perhaps more intuitive for some. It currently takes about 1 hour on a laptop to export data to the OpenNMT format we used in our experiments.

A.2 Detailed responses to reviewer questions

We had two questions from reviewers which we wanted to address in more detail, but would have disrupted the flow of the paper had we included them directly in any of the sections. Both questions are interesting, and worth addressing, we thank the reviewers for them.

A.2.1 Should cross-fold partitions be used?

The short answer is we believe so, but it would take too much time. Whilst we maintain that partitions should not cross season boundaries in this dataset, systems would ideally be evaluated using different combinations of seasons for training, validation and testing. Our selection of 2014-16 for training, 2017 for validation and 2018 for testing was only one possible setup. If we had evaluated with different partition setups we would have perhaps been able to determine if the per-season partition BLEU score in Figure 4 was an anomaly or a generally seen increase. The problem was that our setup for this paper took about 3 weeks of compute time. Testing additional partition setups was therefore not practical in the time frame we had.

A.2.2 What is the performance difference between this resource and the previous one?

This is a tricky question to answer because it would depend on what the resource was being used for, as well as whether raw SQL or the ORM was used. It also depends on the implementation of code which would read the previous JSON format. When using SQL, some queries will be significantly faster than with JSON. For example, for a separate project, when fact checking texts manually, we encountered a sentence like ‘The Wizards came into this game as the worst rebounding team in the NBA this season’. Whilst possible to check with the old JSON format, it would be both slow and difficult. A short SQL query was able to find this information quickly because all keys are indexed. The ORM is inherently slower than raw SQL, being a wrapper layer on top of it. However, given that users can use either SQL, the ORM, or a combination of both (raw SQL within the ORM), we feel we have improved data quality, and ease of use, without sacrificing efficiency. The data is not meant for production systems, it is designed for research. Therefore, provided that data can be generated for experiments within a couple of hours (with the ORM), we feel this is sufficient. By adding more SQL this time would come down, but it would take a little longer to implement the export script. The time taken for data processing is still likely to be much less than the downstream compute time and therefore should not cause unreasonable disruption to any research project which uses it.

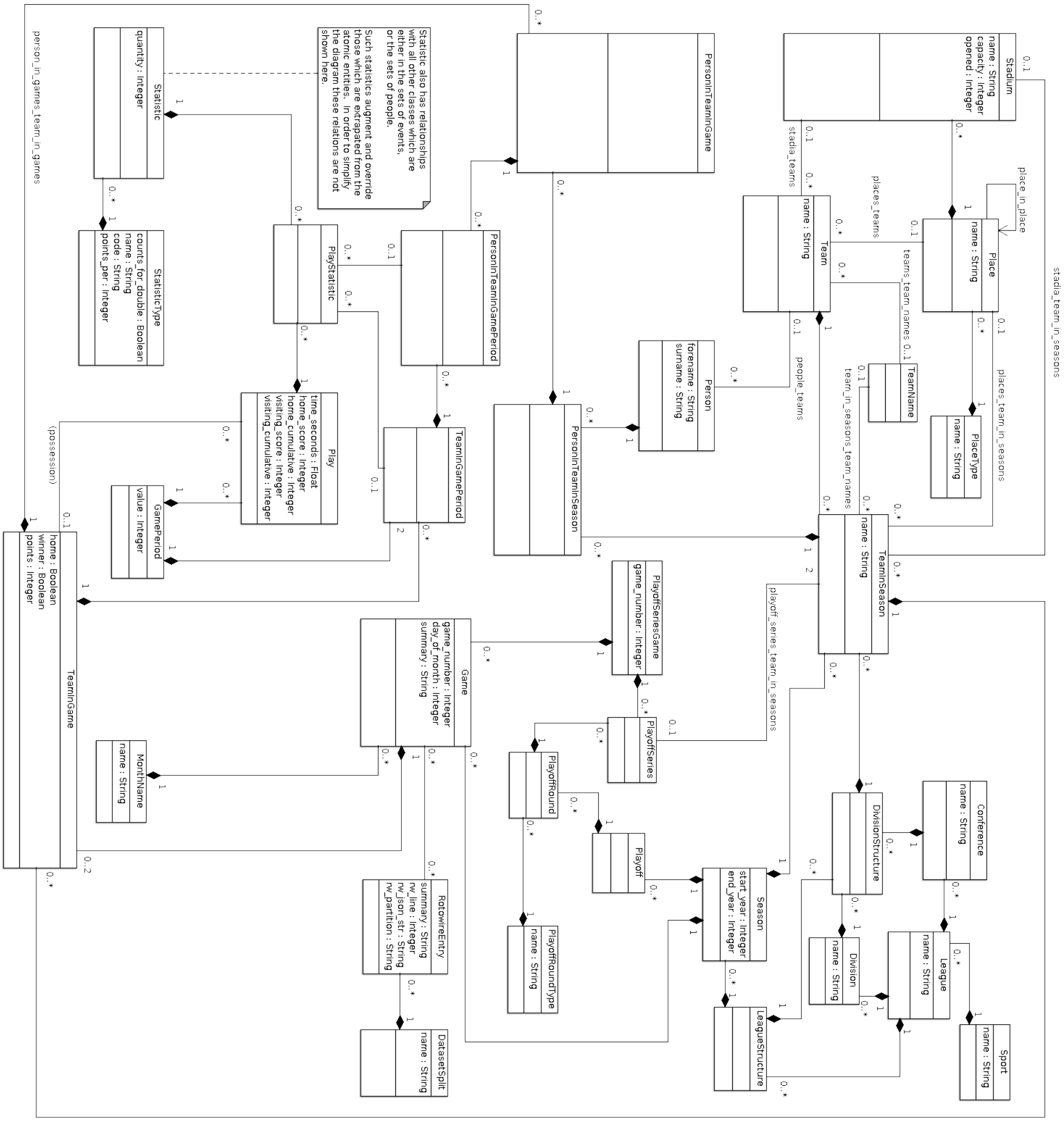


Figure 5: Entities in our database shown as a UML Class Diagram. Classes and named relations are all mapped to individiaul SQL tables.