Data-Driven News Generation for Automated Journalism

Leo Leppänen and **Myriam Munezero** and **Mark Granroth-Wilding** and **Hannu Toivonen**
Department of Computer Science
University of Helsinki
Helsinki, Finland

Abstract

Despite increasing amounts of data and ever improving natural language generation techniques, work on automated journalism is still relatively scarce. In this paper, we explore the field and challenges associated with building a journalistic natural language generation system. We present a set of requirements that should guide system design, including transparency, accuracy, modifiability and transferability. Guided by the requirements, we present a data-driven architecture for automated journalism that is largely domain and language independent. We illustrate its practical application in the production of news articles upon a user request about the 2017 Finnish municipal elections in three languages, demonstrating the successfulness of the data-driven, modular approach of the design. We then draw some lessons for future automated journalism.

1 Introduction

Traditional media companies around the world are confronted by many challenges stemming from the radical digital transformation of the publishing industry. This includes the increase in availability of and access to data, on a scale that prohibits journalists from handling it and using it for news reporting. An essential development in this area is the automation of editorial processes, with capabilities to work with scalable, data-driven content (Linden, 2017). Noticeably, while natural language generation (NLG) as a field has made huge strides in recent years, few NLG systems have been developed to meet the journalistic automation needs of newsrooms (Linden, 2017).

In this paper, we investigate the design and development of NLG systems for generating news. Specifically, we aim to address the question: *How should an NLG system be designed to meet journalistic requirements?*

Based on a review of literature on automation of journalistic processes, we identify a set of six requirements (e.g., transparency, accuracy, modifiability) that should be considered when developing NLG systems for the newsroom. We then describe a data-driven architecture that is modular and to a large extent domain and language independent. We demonstrate the applicability of the design by implementing a real-world automatic news generation system[1] that was capable of producing hundreds of thousands of news articles about the Finnish municipal elections that took place in April 2017, in three languages. Responding to user requests, the system was able to generate hyper-localized news, e.g. on how one candidate fared at a single polling station.

2 Related work

Carlson (2015) defines automated journalism as programs or algorithms capable of converting structured data into publishable news stories without human intervention. A recent review (Linden, 2017) of the state of automated journalism found that, while NLG as a field has taken huge strides, these changes have not so far manifested themselves in newsrooms. While many media companies, such as the Associated Press, Forbes, and the New Yorker, have

[1] https://www.vaalibotti.fi

Proceedings of The 10th International Natural Language Generation conference, pages 188–197,
Santiago de Compostela, Spain, September 4-7 2017.

started to automate production of news content due to the demands and pressures of reduced resources in newsrooms, the majority of news generation efforts have been in domains where structured data is abundant and the domains are well understood, such as weather forecasting (Sripada et al., 2003; Goldberg et al., 1994), weather news summaries (Chen and Huang, 2014), finance (Nesterenko, 2016; Mendez-Nunez and Trivino, 2010; Andersen et al., 1992) and sports (Bouayad-Agha et al., 2012; Theune et al., 2001; United Robots, 2017).

As most systems used in newsrooms are built by private companies, details of their architectures and operation are scarce. From what information is available, the majority appear to be heavily rule- and template-based. For instance, the widely used Wordsmith (Automated Insights, nd) enables journalists to easily build templates for whole stories. Homicide Report (Los Angeles Times, 2007) and Quakebot (Los Angeles Times, 2014) by the Los Angeles Times, use templates with simple, hand-written rules (Young and Hermida, 2015). In contrast, Quill (Narrative Science, nd) uses a data ontology, together with user input, to generate text, relying less on application-specific templates. A development manager from a Nordic news company points out: "It is difficult to create generic solutions, [newsrooms] have to start from scratch for each new case, and relatively little is reusable" (Linden, 2017). Newsrooms use primarily whole-story templates and custom software that is not easy to port across domains and produces news in one language.

NLG systems generally range from the classical type of multi-staged architecture described by Reiter and Dale (2000) to end-to-end architectures, such as recurrent neural networks. For reasons discussed below, we here focus on the pipeline architecture common in earlier NLG work (Reiter, 2007; Mahapatra et al., 2016), consisting of document planning, micro planning, and surface realization stages.

3 Requirements for Journalistic Natural Language Generation

An NLG system for news production must try to meet some if not all requirements that are placed on journalists. From literature on journalism, we identify a set of six requirements that are important in journalism in general and should adequately be reflected in journalistic NLG. These stem from three primary motivations: the journalistic process (requirements for transparency and accuracy), the system itself (modifiability and transferability, fluency) and the application of the system (data availability and topicality).

Transparency While trust in the underlying data is not strictly an NLG problem, trust in the system that transforms the data into an article is a concern. This is related to journalistic transparency and accountability. Transparency has been defined by Deuze (2005) as the "ways in which people both inside and external to journalism are given a chance to monitor, check, criticize and even intervene in the journalistic process." In the case of automated journalism, transparency might include making public the steps taken to process the data, the analysis code, the model or inferences made with it, software used, or data sources (Diakopoulos and Koliska, 2016). This transparency in turn reinforces the goal of media accountability (McBride and Rosenstiel, 2013; Stark and Diakopoulos, 2016). This requirement, however, may be in opposition to the possible business benefits of keeping the algorithmic processes hidden (Diakopoulos, 2015).

For end-to-end (e.g. neural network-based) NLG systems, transparency can become an issue, since their 'black box' design prohibits inspection of the generation process, which makes tracing the decisions of the system often impossible. Traditional pipeline systems are better suited to providing transparency, as they allow the processing to be logged and observed to a high degree and individual decisions to be traced to specific components.

Accuracy Like journalists, a journalistic NLG system must meet basic standards of journalistic accuracy. The produced text must be supported by factual data and should not provide any misleading statements (Wright, 2015; Smiley et al., 2017). It may be appropriate to add a disclaimer on the accuracy or correctness of data (Smiley et al., 2017).

Modifiability and transferability of the system Software modifiability is a desirable characteristic of NLG systems, as well as other software systems

(Oskarsson, 1982). In particular, for journalistic NLG systems, easy modifiability of system components in order to incorporate new knowledge will allow for transferability to other domains. This is especially important for newsrooms that have a declining budget and resources, where it would be desirable for one NLG system to produce news over several domains.

Fluency of output Just as humans, NLG systems are expected to output natural sounding texts that are coherent and fluent (Gatt and Krahmer, 2017). Fluency helps make NLG systems more usable (Oberlander and Brew, 2000), which would help maintain or increase newsrooms' customers' satisfaction.

Data availability Although availability of (structured) data is a requirement for all NLG systems, for newsrooms it affects or dictates a number of areas. For instance, it affects the domains and topicality of generated news and the speed of content production (Dörr, 2016). In addition, the data must be newsworthy and available in good quality for accurate reporting. Frequency of availability further affects how regularly an NLG system can produce news. Thus newsrooms first have to be aware of available data sources before investing in NLG systems.

Topicality of news To ensure user interest, the produced news should be topical, discussing recent events that are relevant to the reader (Dörr, 2016). One way to increase topicality is to focus on locality, covering stories for small, local audiences which currently go unreported (Van Dalen, 2012). This might take the form of tailoring the content, style or even the language of the article (Dörr, 2016; Linden, 2017). Highly localized news needs to be targeted to readers, either by automated means or by allowing the user to interact with the service (Dörr, 2016).

4 A Data-Driven NLG Architecture

We present an architecture for a journalistic NLG system, addressing the above process- and system-oriented requirements. An overview of the design is given in Figure 1. It aims to be modular with respect to application domain, language, and representation of individual facts. We next describe how the design supports these goals, and in the following section describe an instantiation of this architecture.

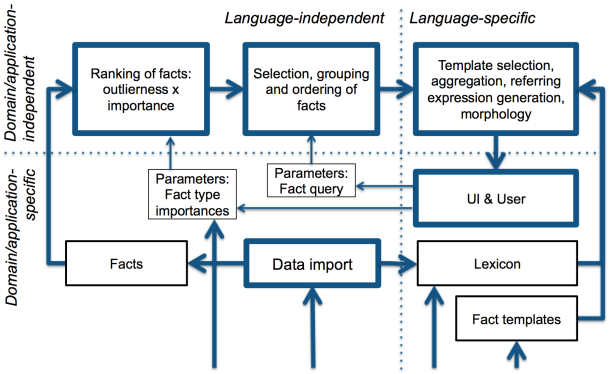

Figure 1: Overview of the architecture. Thick boxes represent software components and thin boxes data structures.

Generic representation of facts Aiming for wide applicability and domain independence, all facts are represented using identical data structures. Inspired by the informal need for news stories to communicate *what* took place, *where* and *who* was involved, we represent a fact as a triplet of *(entity, location, value)*. This minimal set of information could be extended as necessary to cover aspects such as *when*.

To allow explicit handling of different types of entities, locations, and values, each field will in practice need to specify a category or type. For example, the system may distinguish 'person' and 'party' entity types. The triplets thus become sextuplets *(entity_type, entity, location_type, location, value_type, value)*. The domains of types and values are open; the core components are made application independent by allowing such meta-information to be specified later for a particular domain.

Use of Flat Sets of Facts The sets of facts obtained as input data are assumed to have no structure beyond their sextuplet forms. All facts are initially considered equal such that stories could, in principle, express each fact and in any order.

The design uses templates for short natural language expressions. The templates are typically specified at the fact level, which gives the system the possibility to order and aggregate them to create richer sentences or paragraphs.

These assumptions together allow for application independence in core components: there is no need for application-specific data structures or story structures. It is easy to make incremental changes or additions to the possible fact types, adding templates

as necessary without changes to the existing ones.

Data-Driven, Application-Independent Fact Ranking Method A measure of *newsworthiness* is computed for facts as the product of two factors.

Outlierness: evaluates how exceptional a given value is when compared to other values of the same *value_type* within the same location. It is completely data driven and calculated with respect to the interquartile range of the distribution of the values.

Importance: depends on the application and the user. In the design, we parameterize its computation, so that parameters can be expressed for a specific application in a modular fashion and changes to importance can be made at run-time according to the user's interests. The parameters specify weights for different *entity_type*s, *location_type*s and *value_type*s; the overall interestingness is their product.

This parameterization makes it easy to generate highly tailored and localized news, by adjusting parameters in response to, e.g., user profile data or user interaction.

Data-Driven, Application-Independent Selection, Grouping and Ordering of Facts A document plan is created with application-independent methods using the newsworthiness-ranked list of facts, potentially constrained to a particular location or entity by the user. Since no fixed story structures are assumed, the most newsworthy fact is chosen as the *nucleus* – the most important part (Forsbom, 2005) – of the first paragraph, to which other supporting facts, or *satellites*, are added.

Isolation of Language-Specific Aspects Language-specific aspects are isolated in a modular fashion. First, each new language requires fact templates and a lexicon, providing language-specific expressions for entities in the input data, for use in referring expression generation (REG). Second, aggregation, REG, and morphology components must be modified for the language; existing tools and methods can often be used. The templates, realization component, and user interface are the only language-dependent parts of the architecture.

Application-Independent Microplanning Based on Fact Templates The final story is constructed from small templates, typically each representing one fact. The templates are the only application-dependent input to the process aside from the data itself; the proposed design allows microplanning (template selection, aggregation, and REG) and surface realization (morphology) to be performed in an application-independent manner.

5 A Data-Driven Election News System

5.1 Finnish Municipal Elections

The architecture presented above was implemented and applied to the Finnish municipal elections of 2017. These occur every four years and are a large political event in Finland, comparable to presidential and parliamentary elections in voter turnout.

The election results, as released immediately after the first count by the Finnish Ministry of Justice (MoJ), offer ample opportunities for automated news production. The data is made available in a machine-readable format. The files include the results for each party on the level of the whole country, each of the 13 electoral districts, 311 municipalities, and 2,012 polling stations. For each of the 33,316 candidates, the data includes details of their success in their own municipality (each candidate is only votable in a single municipality) and all of the municipality's polling stations.

In total, 727,404 valid combinations of locations and entities (party, candidate, or none) are possible. Each of these combinations corresponds to a distinct topic for a news story, satisfying the goal of producing news at varying levels of locality.

The Finnish context further calls for multilinguality. Finland is bilingual, with both Finnish and Swedish being official languages. This situation, not uncommon in other countries with several official languages or minority languages, exemplifies the modifiability and transferability requirements discussed in Section 3. With international audiences in mind, it was also desirable to produce articles in English.

5.2 Implementation

The design of the election news system follows Figure 1. Software components corresponding to the top part of the figure are generic, and any information about the election domain is provided to them as parameters or data at run-time. The bottom part

of the figure is domain and application specific.

Fact Types Components in the lower part of the figure work with specific fact types, while the upper part treats them as abstract symbols. In the election system, fact types consist of three components: the types of the entity, the location and the value concerned. Entity has two types: party and candidate. Location has four types: the whole country, electoral district, municipality, and polling station.

Value types are more varied. They capture numerical information such as the absolute number of votes received by an entity in a location, the percentage of votes received, and the number of seats obtained (distinction A in computing importance, below). For each of these types of information, we also have the corresponding numbers from the previous municipal elections, and thus also the change in each of them (B). In addition, for each of these nine numbers, we can compute ranks, e.g., who received most votes, or who won most new seats (C). We can also compute reverse ranks, which are interesting especially for changes: who lost most votes or most seats. With the addition of some facts providing background information about candidates and the removal of some redundant types, we have a total of 14 value types per candidate and 21 per party, for each applicable location.

Data Import Data is imported from the MoJ online system with custom-built software. Facts are extracted and derived according to the sextuplet schema *(entity_type, entity, location_type, location, value_type, value)*. Slightly over 10 million facts were obtained.

Lexicon Finnish and Swedish lexicons are automatically extracted from the MoJ dataset at data import, providing natural language expressions for entities and locations. In the English lexicon we use the Finnish names. For this application, we had no use for lexical entries of values, which are largely numeric or boolean.

Some additional entries were specified for each language: alternative short names for parties, as well as pronouns and other generic referring expressions used for entities and locations, such as "the party" and "it" for parties.

Templates We use a simple templating language. A template expression consists of (1) a sentence expressed in natural language with slots that can be filled with information from facts, such as "`{entity} won {value} new seats in {location}`", and (2) conditions on what facts the template can be used for, such as "`value_type = nr_of_seats, value > 0`". Slots are optionally associated with cases to instruct the morphological component.

Multi-linguality is supported by allowing expressions in different languages to be specified within the same template. This allows different languages to share the conditions and logic of templates. One template can also have multiple expressions for the same language.

We supplied the system with 100 templates for Finnish, averaging 5.7 tokens; 102 for English, averaging 9.4 tokens; and 70 for Swedish, averaging 8.3 tokens. A separate, smaller set of templates was used for generating headlines. Templates were written by journalists who are native speakers of the respective languages.

Fact Type Importance Value types have a compositional structure, corresponding to distinctions A, B and C in Fact Types above: they talk about the absolute or relative number of votes or of seats (A), of the current or previous election results or the change (B), and possibly about the ranking or reverse ranking of the values (C).

From a news reporting perspective, some of these value types are more important than others. For instance, the number of seats is important because it directly reflects the power of a party, and changes are usually interesting to readers; on the other hand, the absolute numbers of votes are mostly minutiae.

The importance of any given value type is defined as the product of factors reflecting distinctions A, B, and C. This allows developers to tune their view of fact importance in a modular way. In future, the same tuning could also be performed, either directly or implicitly, by the user at generation time.

In the same way, factors are specified for different entity types and location types and included in the importance of a fact. The overall newsworthiness of a fact is the product of the fact type importance and of the outlierness of the value (cf. Section 4).

Fact Selection, Grouping and Ordering To support locality and selectivity of news, the system allows users to make queries based on an entity and a location. The system then extracts only those facts that have the specified location and entity, letting the user specify what is relevant for them.

Among the extracted facts, the system picks the most newsworthy as a nucleus. Satellite facts that share the nucleus' location and entity are then selected, ordered by their newsworthiness. Satellites are added until the paragraph is considered full (in our case, consisting of five facts) or the newsworthiness drops below 20% of the most newsworthy fact in the story.

Nuclei for subsequent paragraphs are selected in order of newsworthiness, but giving preference to facts with a different location or entity to the previous nucleus. Paragraphs are added in this way until either a maximum number (five) is reached or there are no newsworthy candidate nuclei remaining.

Additionally, the system handles context changes. If a fact talks about a new location, this change in context will need to be expressed in the resulting text. If no suitable template that expresses the location exists, the fact is disallowed at that point in the document plan.

Template Selection Realization of facts in a given language is performed by instantiating matching templates to obtain initial sentences. Each fact in the document plan is associated with a template. If there are several choices meeting all requirements, one is picked at random.

Aggregation The aggregation component simply checks whether two subsequent templates contain a common prefix of one or more tokens. If they do, and the first sentence is not already a product of aggregation, the two sentences are joined together by a conjunction (e.g., "and"), with the common prefix removed from the second sentence.

Referring Expression Generator The REG consults the lexicon for names of entities. When an entity is first encountered in the document plan, the full name (e.g. "Kansallinen Kokoomus") is always used. On subsequent encounters, the short name (e.g. "Kokoomus") or the pronoun form is used depending on whether the previously mentioned named entity is a different entity or not. Instead of the pronoun form, an equivalent general expression (e.g. "the party") can be also used.

Morphology To produce morphologically correct forms of filled slots, a few simple rules suffice for Swedish and English. Finnish morphology is significantly more complex, with 15 cases. We use Omorfi (Pirinen, 2015) to perform inflection, with some additional rules for uncommon names.

User Interface The system is accessed via a web interface. The user can specify a query via an entity or a location, or both, or neither, and the system produces the corresponding story. After the generated story, headlines linking to random recently read articles are displayed.

5.3 Deployment

The web application was launched at the night of Finnish municipal elections in April 2017 on a website not associated with any news organization or university, at `https://www.vaalibotti.fi`.

During its first day of operation, it saw 398 unique visitors accessing 573 unique news stories. Out of these, 343 (60%) stories were viewed only by one user each. This indicates a need for a high degree of tailoring of news stories in this application.

The system and the reports it generates were made freely available for reuse. At least one Finnish newspaper published stories generated by the system on their website, as-is without any editing.

An example of the system's English language output is presented in Figure 2. A formal quantitative evaluation of the output is outside the scope of this article: we concentrate here on how the constraints set by the journalistic domain influence system design. However, we make some brief, subjective and informal remarks of the produced reports.

Overall, the generated news is expressed in clear language. The few linguistic errors are mostly due to the complexity of Finnish morphology. The texts are fluent, but not always as fluent as human-written texts. There seem to be two main reasons for this: fact ordering sometimes is suboptimal (e.g., the second-last sentence in Figure 2), and the simple aggregation method sometimes combines sentences in unnatural or misleading ways (e.g., the last sentence in Figure 2; "17.5%" refers to current elections, not

Perussuomalaiset drop most seats across Finland

Perussuomalaiset dropped the most council seats throughout Finland and got 3.5 percentage points fewer votes than in the last municipal election. Perussuomalaiset decreased their voter support by the greatest margin and has 769 seats in the new council. The party secured 4th most council seats.

Throughout Finland, Vihreä Liitto secured the largest gain in council seats. The party increased their voter support by the greatest margin and got 3.9 percentage points more votes than in the last municipal election. 12.4% of the vote went to the party. The party got 319440 votes.

Suomen Keskusta lost 254 seats and the second most council seats. The party has 2823 seats in the new council and won the most council seats nationwide in the previous election. The party secured 17.5% of the vote and had 3077 seats in the previous council.

Figure 2: Excerpt of the system's output. The user has requested an overview article on country-wide results without any specific candidate or party as focus.

the previous ones). These results are unsurprising given the simplicity of the corresponding components in our implemented system.

Fact ranking, selection and grouping work well. Stories generated for different queries exhibit a wide range of different story structures and contents, reflecting the corresponding election results in a natural way. The balance of story diversity and quality is in our view striking, given that stories are composed of sentence-long components.

6 Discussion

Reflecting on the requirements for journalistic NLG systems described in Section 3, in this section we discuss, based on our experiences, how the presented data-driven NLG design manages to satisfy them.

Transparency A journalistic NLG system should be transparent at least to the editorial staff of the newsroom, to give it accountability and ensure trust. The simple and modular architecture we described is easier to understand than a large and complex system, not to mention black-box architectures. It allows journalists to observe the generation processes in detail via fine-grained logging, and any decisions made by the system can be explained if needed. For instance, the reason for including any specific fact in the story can be traced back to nucleus-satellite relations in the story structure and to the weights of fact types given as parameters.

The implemented election system is, however, not as transparent as systems built around a story template with explicit, human-written rules of how to apply the templates. There seems to be a trade-off here between general-purpose architectures and custom-built ones. However, in the long run, it may be easier for journalists to understand the principles of one generic system rather than many manually built ones.

Accuracy The accuracy of a news story is paramount and it is critical that any journalistic NLG system truthfully describes the underlying facts. In the proposed design, it is clear that facts are not modified or faked. Two questions remain: are the "right" facts selected, and are they shown in a manner that accurately conveys their meanings.

Fact selection is based on parameters that adjust fact type importances as well as the data point outlierness. It is possible – or even likely – that the result does not correspond exactly to what a journalist would report, but we can expect disagreement among journalists too on this. The benefit of transparency, and here especially of the explicit fact type importance parameters, is that they can be discussed and modified in the light of a system's behaviour.

Fact templates need to be carefully written to not misrepresent any situation. For example, the template "`{party} got the most votes in {location}`" can be technically true even when two distinct parties were tied for the most votes received, but the reader would be misled to think that there was one winner.

Aggregation of templates by the simple, surface-level technique our system uses, can also produce misleading sentences. For example, aggregating "X got most votes", and "X got 5 seats in the previous elections" into "X got most votes and 5 seats in the previous elections" changes the meaning of the first text snippet. Avoiding this sort of error requires a more sophisticated aggregation component with, at least, an awareness of syntactic structure – a well-studied task in NLG (Dalianis, 1999).

Modifiability and transferability of system Our case study shows that it is possible to design an NLG system so that the domain-specific, language-

specific, and general subsystems are separated. Transferring a system to a new language or a new domain then entails only localized changes in the respective components. In the multilingual election application we added English last, and the additional effort required was minimal, consisting mostly of translating template sentences to English.

The design and its successful implementation in the election system also demonstrate that language- and domain-independent software components can have a central role in content determination and document planning, and any information that is dependent on either the language or the domain is provided as parameters or small extensions.

Modularity also opens up interesting avenues for automation of parts of the architecture that were built by hand in our election system. Modularity with respect to fact types, for instance, allows for the possibility of automated acquisition of templates.

Fluency of output The requirement for the fluency of a journalistic NLG system is dependent on the intended audience and use of the system. If the intention is to produce stories to subscribers (rather than drafts to journalists, or short fragments of text), the system must produce near-flawless text. In a data-driven architecture like ours, it is safer to use simpler sentence structures than more complex, aggregated structures, for reasons discussed above.

Based on our experience, a large balancing act exists between transferability of the system and the fluency of the produced text. Long and complex templates are less transferable to different domains or languages, while general rules end up not producing the kind of idiomatic language one would expect from a human journalist.

Data availability and topicality of news These requirements have more to do with the application domain than the system or processes. Since the focus of this paper is on the latter, we only make some general remarks.

The data source used in this case study was an example of a rarely published but large data set. Other domains might have data sources that produce smaller amounts of data near-constantly. Evaluating whether the amount and frequency of published data is enough in any particular scenario to make automation worthwhile is an important step when considering potential applications (Reiter and Dale, 2000).

7 Conclusions

Automated journalism is an interesting and understudied application area for NLG. We examined the needs of journalistic systems and extracted six key requirements. We proposed a data-driven, modular architecture to address the requirements and described its instantiation in news generation for municipal elections.

The architecture allows for the implementation of an NLG pipeline using application- and language-independent software components, parameterized with application-specific information about fact types. The design as well as the election system successfully demonstrate that the data-driven, modular approach is able to select and structure content in an accurate and transparent manner and output fluent text. The high modularity further made it straightforward to produce text in three languages as well as tailor the text based on user request.

In future work, we will conduct a more formal evaluation of the texts output by the election system. Whilst the support for multiple languages validates some of our design decisions, it is also crucial to test the proposed design principles by transferring the system to new domains. Potential improvements during this process include more versatile fact types (e.g., adding a time element to facts) and identifying generic alternatives for the flat fact structure (e.g., stories that have a causal structure), possibly using existing, general formalisms, such as Abstract Meaning Representations. Finally, we aim to apply machine learning to specific components within the architecture, such as automatic acquisition of templates.

Acknowledgements

This work is part of the “Immersive Automation” research project which has received funding from Tekes, the Finnish Funding Agency for Innovation, and from companies.

References

Peggy M Andersen, Philip J Hayes, Alison K Huettner, Linda M Schmandt, Irene B Nirenburg, and Steven P

Weinstein. 1992. Automatic extraction of facts from press releases to generate news stories. In *Proceedings of the third conference on Applied natural language processing*, pages 170–177. Association for Computational Linguistics.

Automated Insights. n.d. Wordsmith. https://automatedinsights.com/wordsmith. Accessed: 2017-04-07.

Nadjet Bouayad-Agha, Gerard Casamayor, Simon Mille, and Leo Wanner. 2012. Perspective-oriented generation of football match summaries: Old tasks, new challenges. *ACM Trans. Speech Lang. Process.*, 9(2):1–31.

Matt Carlson. 2015. The robotic reporter: Automated journalism and the redefinition of labor, compositional forms, and journalistic authority. *Digital Journalism*, 3(3):416–431.

Shyi-Ming Chen and Ming-Hung Huang. 2014. Automatically generating the weather news summary based on fuzzy reasoning and ontology techniques. *Information Sciences*, 279:746–763.

Hercules Dalianis. 1999. Aggregation in natural language generation. *Computational Intelligence*, 15(4):384–414.

Mark Deuze. 2005. What is journalism? professional identity and ideology of journalists reconsidered. *Journalism*, 6(4):442–464.

Nicholas Diakopoulos and Michael Koliska. 2016. Algorithmic transparency in the news media. *Digital Journalism*, pages 1–20.

Nicholas Diakopoulos. 2015. Algorithmic accountability: Journalistic investigation of computational power structures. *Digital Journalism*, 3(3):398–415.

Konstantin Nicholas Dörr. 2016. Mapping the field of algorithmic journalism. *Digital Journalism*, 4(6):700–722.

Eva Forsbom. 2005. Rhetorical structure theory in natural language generation. *Téléchargé le*, 10(08):2008.

Albert Gatt and Emiel Krahmer. 2017. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *arXiv preprint arXiv:1703.09902*.

Eli Goldberg, Norbert Driedger, and Richard I Kittredge. 1994. Using natural-language processing to produce weather forecasts. *IEEE Expert*, 9(2):45–53.

Carl-Gustav Linden. 2017. Decades of automation in the newsroom: Why are there still so many jobs in journalism? *Digital Journalism*, 5(2):123–140.

Los Angeles Times. 2007. Homicide report. http://homicide.latimes.com/. Accessed: 2017-04-07.

Los Angeles Times. 2014. Quakebot. http://www.latimes.com/local/lanow/earthquake-27-quake-strikes-near-westwood-california-rdivor-story.html. Accessed: 2017-04-07.

Joy Mahapatra, Sudip Kumar Naskar, and Sivaji Bandyopadhyay. 2016. Statistical natural language generation from tabular non-textual data. In *INLG*, pages 143–152.

Kelly McBride and Tom Rosenstiel. 2013. *The new ethics of journalism: Principles for the 21st century*. CQ Press.

Sheila Mendez-Nunez and Gracian Trivino. 2010. Combining semantic web technologies and computational theory of perceptions for text generation in financial analysis. In *Fuzzy systems (fuzz), 2010 IEEE international conference on*, pages 1–8. IEEE.

Narrative Science. n.d. Quill. http://www.quillcontent.com/. Accessed: 2017-04-07.

Liubov Nesterenko. 2016. Building a system for stock news generation in Russian. In C Gardent and A Gangemi, editors, *Proc. of the 2nd International Workshop on Natural Language Generation and the Semantic Web (WebNLG 2016)*, pages 37–40, Stroudsburg, PA. ACL.

Jon Oberlander and Chris Brew. 2000. Stochastic text generation. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 358(1769):1373–1387.

Östen Oskarsson. 1982. *Mechanisms of modifiability in large software systems*. Linkping Studies in Science and Technology. Dissertations,0345-7524; 77.

Tommi A Pirinen. 2015. Omorfi free and open source morphological lexical database for Finnish. In *Proceedings of the 20th Nordic Conference of Computational Linguistics, NODALIDA 2015, May 11-13, 2015, Vilnius, Lithuania*, number 109, pages 313–315. Linköping University Electronic Press.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Studies in Natural Language Processing. Cambridge University Press.

Ehud Reiter. 2007. An architecture for data-to-text systems. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, pages 97–104. Association for Computational Linguistics.

Charese Smiley, Frank Schilder, Vassilis Plachouras, and Jochen L Leidner. 2017. Say the right thing right: Ethics issues in natural language generation systems. *EACL 2017*, page 103.

Somayajulu Sripada, Ehud Reiter, and Ian Davy. 2003. SumTime-Mousam: Configurable marine weather forecast generator. *Expert Update*, 6(3):4–10.

Jennifer A Stark and Nicholas Diakopoulos. 2016. Towards editorial transparency in computational journalism. *Computation + Journalism Symposium*.

Mariët Theune, Esther Klabbers, Jan-Roelof de Pijper, Emiel Krahmer, and Jan Odijk. 2001. From data to speech: a general approach. *Natural Language Engineering*, 7(01):47–86.

United Robots. 2017. Rosalinda for sports. retrieved on 24th April 2017, `http://www.unitedrobots.se/produkter-1/`.

Arjen Van Dalen. 2012. The algorithms behind the headlines: How machine-written news redefines the core skills of human journalists. *Journalism Practice*, 6(5-6):648–658.

Alex Wright. 2015. Algorithmic authors. *Communications of the ACM*, 58(11):12–14.

Mary Lynn Young and Alfred Hermida. 2015. From Mr. and Mrs. outlier to central tendencies: Computational journalism and crime reporting at the Los Angeles Times. *Digital Journalism*, 3(3):381–397.