http://dx.doi.org/10.7592/EJHR2017.5.4.rayz

European Journal of
Humour Research 5 (4) 169–178
www.europeanjournalofhumour.org

Computational humour and Christie Davies' basis for joke comparison

Julia Taylor Rayz
Purdue University
jtaylor1@purdue.edu

Abstract

While historically computational humour paid very little attention to sociology and mostly took into account subparts of linguistics and some psychology, Christie Davies wrote a number of papers that should affect the study of computational humour directly. This paper will look at one paper to illustrate this point, namely Christie's chapter in the Primer of Humor Research. With the advancements in computational processing and big data analysis/analytics, it is becoming possible to look at a large collection of humorous texts that are available on the web. In particular, older texts, including joke materials, that are being scanned from previously published printed versions. Most of the approaches within computational humour concentrated on comparison of present/existing jokes, without taking into account classes of jokes that are absent in a given setting. While the absence of a class is unlikely to affect classification—something that researchers in computational humour seem to be interested in—it does come into light when features of various classes are compared and conclusions are being made. This paper will describe existing approaches and how they could be enhanced, thanks to Davies' contributions and the advancements in data processing.

Keywords: Computational humour, joke comparison, Christie Davies, computer science.

The computer's (in)ability to process humour has attracted more and more attention recently. Researchers around the world are trying to classify what is funny and what is not funny, with an idea that sophisticated algorithms might be good at the task. Not surprisingly, these algorithms (see Taylor 2017 for a review) are not good at the task, but there are reasons why they cannot be good, for a majority of papers that are published. First of all, computational humour, if done correctly, demands for researchers to be familiar not only with the state of the art of whatever mode of communication they are processing, but also with humour research per se, which, as we know, is multidisciplinary. Second, it requires a good dataset that a computer can use to 'learn' from positive and negative examples. It is an open question whether a computer can learn/detect/understand humour without 'some'—somewhere between none and all—knowledge of the world.

This paper is not going to address the state of the art of any (type of) mode of communication that is suitable for computational humour (for various stages of the state of

the art see Ritchie 2001; Hempelmann 2008; Miller *et al.* 2017; Taylor 2017). Rather, it will address the question of a 'good' dataset and, what is referred to in linguistics and Artificial Intelligence as "world knowledge": an often argued about (Hobbs 2009) approximation of prototypical events and objects that people can interpret, understand and about which they are able to reason. This paper will start from the assumption that it is much easier for computational humour to address jokes rather than increasingly popular sarcasm and/or irony detection because, arguably, the former one does not require an understanding of as much context as the latter ones. It will also use Christie Davies' (2008) definition of jokes: "Jokes are short, compact units which in most cases can be quickly understood and enjoyed by the broad masses of the workers, peasants and petit-bourgeoisie alike.…Jokes are numerous and do not have authors; they are invented by, improved by and circulated among large aggregates and networks of individuals. Jokes are a true spontaneous product of the imaginations of and a good reflection of the tastes of ordinary people." And, they appear online as well as in books, which makes them an easy source for processing by the computer. The rest of the discussion is about jokes; however, it can be applied to other forms of humour.

Why should it take an entire paper to talk about a dataset, and why does it matter whether it is good or bad? Intuitively, we know the answer as to why it matters whether it is good or bad: garbage in, garbage out. If we don't have enough examples for classification of (good) jokes and non-jokes, then an algorithm for joke detection will show nonsense, even though the classification will be complete. However, there is a deeper question to be asked here, if we are discussing a dataset: is it only useful for a classification of jokes/non-jokes, or are we trying to learn something from it? Hopefully, the answer is yes to the latter, which leads to the next questions:

- When we classify (good) jokes and non-jokes, what is our basis of classification?
- Are we trying to classify jokes based on surface features, such as similar words, or is it more important to look at deeper representation and implicit reasoning that is happening in the joke?
- Is it enough to classify something as a joke/non-joke, or are we trying to cluster similar jokes together?
- Do we need non-jokes in a corpus, or are we willing to ignore the mechanisms that lead to humour and rather concentrate on the topics about which jokes are made?

There are also questions about a composition of the dataset:

- How complete should it be? In other words, what kind of jokes should go into a corpus?
- If a corpus is of recent jokes—whatever recent means—should jokes only relating to recent events be included?
- Should we mine the web and existing joke sources for jokes related to these events and try to insert all of them into a corpus or should it be a representative sample?
- Should we limit the events to a particular geopolitical and/or language region, or should we include everything around the world that we may find, given that our computational processing supports the language of a given joke?
- Should we assume that the lack of jokes in the corpus implies that such jokes do not exist?

Since any corpus, no matter how complete it is, should address the question of what we are trying to learn from a corpus, we will start our discussion with this set.

What does one want to learn from a joke corpus, if anything?

The first, and most likely only, question to be answered by many computational humour papers is: is a text a joke or a non-joke? There are many ways of answering this question, ranging from theories of humour (for text based humour, you are likely to start with an Incongruity-based family of theories/methodologies) to approximating the 'joke-ness' of a text to its funniness (the approximation is flawed, but, nevertheless, is often used). If one is dealing with a corpus of collected jokes, it is reasonable to assume—provided that the corpus was carefully constructed (see Section 2)—that at a point at which a joke has originated, it was intended to be funny.

Approaches to computational detection of humour range from using words as features for joke detection to attempting to find scripts that are described in the SSTH (Raskin 1985) family of theories of humour. Detection of scripts may be an easy task for a human, but it is a big challenge for a computer that, for the most part, lacks the background knowledge that is assessable to a human. Information that is needed to fill the unstated gaps in a joke is usually missing from its artificial brain. It is usually possible to find some more or less trivial overlaps and oppositions within a text—a necessary component for humour, according to SSTH, GTVH (Attardo & Raskin 1991), and OSTH—but it is difficult to find a salient opposition and overlap that leads to a joke. These trivial overlaps and oppositions that a computer is capable of finding are likely to be not noticed by a human being. Thus, a set of possible scripts (with overlaps and oppositeness) is greatly reduced in human-level understanding of a joke as compared to a computational understanding of it.

It is tempting to assume that if a system can flawlessly classify whether something is a joke or not, it captured the mechanism of humour. An approximation of this assumption could then be something like this: if a system classifies a large number of texts (jokes or non-jokes) correctly, it captured at least some components of the humour mechanism. Unfortunately, for this type of reasoning to hold, the (partial) mechanism that is captured has to hold in general, not just for the corpus at hand. In other words, if a system determined that X correlates with humour in a corpus A, one should check if X correlates with humour outside of a corpus A. In other words, we are looking for negative example: if for a large number of cases outside A, X does not correlate with humour, one should re-examine the validity of X correlation.

This approach is not that different from methodology of comparative research described by Christie Davies (2008): "fish in the morning, hunt in the afternoon and compose in the evening." The fishing part addresses a composition of a dataset: "First, there are simple fishing trips in which the researcher records or finds, listens to or reads several hundred jokes from different sources and kinds of sources" (Davies, 2008). The hunting part is most relevant for the verification of whether X should be listed as a humour feature: "Second, there are planned searches in which he or she is looking for particular sets of jokes or the absence of a particular set of jokes in order to test a hypothesis based on his or her own or on other people's research" (Davies, 2008). Sadly, this hunting component is most often missing from computational humour works.

Another aspect that one may look for in a humour corpus is a list of humour scripts per se, in all their granularity (various levels of details). These various levels of details may lead to a hierarchy of scripts that overlap in much of the information in jokes but not all. For example, are all dumb/smart scripts the same? Is it possible to classify the differences and similarities in a meaningful manner, using a given collection of jokes? The list of scripts and the hierarchy itself, of course, will depend on a composition of a dataset, but there may be interesting similarities between scripts at different levels of a hierarchy that could be captured from the semantic comparison of jokes (and confirmed by humour theories). The similarities

in the details of the scripts should be considered both within information that is present as well as the information that is absent but could have been in all these scripts. From a computational point of view, it is crucial to understand what can be dropped from a joke—for the purpose of a distant future when a computer would be able to compose a joke from scratch.

If one is to look at such explicit and implicit similarities of scripts, clustering similar jokes together is crucial. From a theoretical point of view, we are looking at a chicken and an egg problem: from a GTVH point of view, the jokes are most similar if they have the highest number of matching knowledge resources[1]. However, GTVH assumes that the scripts (as well as their logical mechanism) are already detected. For our purposes, however, we wish to learn scripts from a corpus, so we cannot make the same assumption.

Interestingly enough, we can look at Christie Davies's writing to find a solution to the problem (shamelessly taken out of context): "Most good joke-tellers do not memorize jokes. They simply remember the punch-line, the theme of the joke and possibly a particularly good jab line and then reinvent the story each time it is told." This means that instead of looking for similar scripts, one can simply look for similar punchlines and cluster jokes with similar punchlines together.

This answer to the clustering approach, however, does not solve a chicken and an egg problem above because the jokes within the cluster should still be further differentiated using GTVH. There are at least three reasons for a need of this differentiation. First, jokes about different targets may have the same punchline but slightly altered situation, scripts, etc. Two, at least theoretically, it is possible that our speaker heard the joke somewhere, found two different scripts from what was intended by the original teller but that are still compatible with the punchline, and is now telling the joke accommodating his scripts in the setup. Three, jokes with similar punchlines may have different information omitted, and it is this omitted information may affect the level of details that are explicitly evoked in a given script.

As an example of the third point, consider the following jokes:

(1) Joey-Jim was tooling along the road one fine day when the local policeman, a friend of his, pulled him over. "What's wrong, Seamus?" Joey-Jim asked. "Well didn't ya know, Joey-Jim, that your wife fell out of the car about five miles back?" said Seamus. "Ah, praise the Almighty!" he replied with relief. "I thought I'd gone deaf!"[2]

(2) A little old Irishman gets pulled over by a policeman, who says, "Sir? Do you realize your wife fell out of the car about a mile back?" The old fella replied, "Oh, thank Christ. I thought I'd gone deaf!"[3]

(3) A man who had a little too much to drink is driving home from the city one night, and of course, his car is weaving all over the road. A cop pulls him over. "So," says the cop to the driver, "where have ya been?" "Why, I've been to the pub of course," slurs the drunk. "Well," says the cop, "it looks like you've had quite a few to drink this evening." "I did all right," the drunk says with a smile. The cop says, "Sir, do you realize that your wife fell out of the car several miles back?" "Oh, thank heavens," sighs the drunk. "For a minute there, I thought I'd gone deaf!"[4]

(4) An Irishman who had a little too much to drink is driving home from the city one night and, of course, his car is weaving violently all over the road. A cop pulls him over. "So," says the cop to the driver, "where have ya been?" "Why, I've been to the pub of course," slurs the drunk. "Well," says the cop, "it looks like you've had quite a few to drink this evening." "I did all right," the drunk says with a smile. "Did you know," says the cop, standing straight, and folding his arms across his chest, "that a few intersections back, your wife fell out of your car?" "Oh, thank heavens," sighs the drunk. "For a minute there, I thought I'd gone deaf."[5]

All four versions are posted on the internet, with a very similar punchline. The setup, however, varies between the versions, with (2) having most of the information omitted. It is still possible to reconstruct the main scripts of the joke, and the main scripts in (2) are practically identical to those of (1), (3), (4). It could be argued that (1) has a somewhat different LM than (2)–(4) as (1) explains that a policeman is a friend of a man who lost his wife five miles back, and thus, there might be an explanation why he was not stopped right away. Joke (3) is a curious combination of (4) and (2), being much more similar to (4), but with the wife falling out of a car several miles back even though the events happen in a city. Joke (3) somewhat corrects this seeming clash by replacing several miles with several intersections. From a computational perspective, there is still an unresolved issue of where the event happens in (3) and (4), since a man is driving home from a city. However, we will throw it into "part of the incongruity that doesn't need to be resolved" pile, for the time being.

Our next point of comparison is the Target, one of the six Knowledge Resources of GTVH. What is of interest in these versions is that that (3) never explicitly mentions that the man is Irish. It is not difficult for a person to hypothesize that the joke is about Irish drunk, but a computer is likely to add others to the list—since the only clue that is computationally available is that the event is happening in a geo-political region that has pubs. Similarly, (1) does not explicitly mention that Seamus or Joey-Jim are Irish, although, the names may serve as a clue. If we are to analyse targets of these two jokes, we have an implicit but easily resolved target in (1), an implicit but resolved with more difficulty target in (3), and explicit targets in (2) and (4). Smoothing these differences—algorithmically—to such a low degree that they are almost invisible in human-based GTVH analysis is quite a task without adding any more difficulties. Unfortunately, they are always there.

Let us consider two more texts, taken from Davies (2004):

> (5) Why does an elephant wear red socks?
> Because his yellow ones are in the wash.

> (6) On a visit to the Zoological Gardens in Dublin, Mrs O'Riley noticed that one of the elephants was wearing red socks. ''Tell me, Brendan,'' she said to Mr. Behan, the keeper of the elephant house, ''why is that elephant wearing red socks?'' ''Well,'' replied Mr. Behan, ''you see his yellow ones are in the wash."

Davies notes that (5) "live[s] in a strange half-way house between a true riddle joke (in which there is an identifiable, if irrational, link between the punch line and the question that proceeded it) and a meaningless children's riddle, where the relationship of question to answer is entirely arbitrary" (375), while (6) "is a real joke with a discernible script opposition (SO) centred around the notion of dumbness (i.e., reasonable/ dumb) and a target (TA), the Irish. The logical mechanism (LM) has been rendered clearer, the joke's situation (SI) is now clearly defined in terms of place, characters, and objects, and the narrative structure has changed from that of a riddle to that of a story" (376).

From a computational perspective, we are faced again with two texts that have the same punchline, but have somewhat different characteristics, namely, the narrative strategy, which nicely justifies the omissions in (5) that exist in (6), but, nevertheless, one has to address them in a computational comparison. What is more problematic is that a computer, unlike a human, will see at least one more pair of scripts in (6)—in addition to dumb/smart—that are so obvious that may be ignored by people: elephants don't wear socks. Moreover, this pair of scripts is also present in (5), unlike the dumb/smart pair, and thus creates a better basis of comparison, at least from a computational perspective. It should be noted that the description of dumb/smart opposition is taken out of context here, however another quote is timely: "An

identically worded joke can work with quite different implications in many different situations and be used in many different ways" (Davies 2008).

What these six jokes demonstrate is that even within sets with an identical punchline, there is quite a bit to do computationally before they can be compared according to theoretical foundations. What five of six jokes hint at is that it is possible to compare situations between jokes, given the same targets and high-level pairs of scripts, and determine what higher-level situations are at play (as an additional GTVH knowledge resource). Are these scripts/target combinations applicable to any situation, or is there a subset where they are no longer relevant? It is these "observations," based on a corpus, that let the computer come up with rules of when some jokes could be augmented for a particular situation, given a particular target, based on similar jokes that are known to the system and, eventually, could be (appropriately) delivered in human-computer conversation

Why bother with computational joke classification

When computational "detectors" of humour first came out, the task was simple: given a text, determine whether it should be classified as a joke/humorous/funny or the opposite. Both knowledge-based and machine learning approaches have been used with somewhat similar results: shallow features were detected. In the case of machine learning, these were the features based either on certain words, polarity of certain words, parts of speech, or basic semantic relationships such as synonymy or antonymy. In the case on knowledge based detectors, the number of semantic relationships were greater, but, very rarely did they venture outside of the semantic roles[6]. What happens with these classifications: the computer was given a shallow pattern and asked to make a determination of where a text belonged without making any comparisons between texts within the same category. Essentially, it was given an equation and asked to find a value of a function for a number of arguments.

If that is all that a computational classification is capable of, there is absolutely no use for it from a point of view of humour research because most of the time the patterns that are found are relevant to a given dataset only (see Taylor 2017 for comparison of studies with similar datasets that produced close to opposite results). However, it is possible to do much more with classification than shallow comparison, especially if the texts are classified not only as A and not A, but if the nature of classification is explored. To begin with, let us assume that there are infinitely many jokes (not all of them accessible in written form). Let us further assume that two jokes are similar if they share a number of humour-theoretic features—for example, GTVH knowledge resources—and these features determine the actual value of similarity. Let us further assume, that two jokes that do not share a single feature of any of the knowledge resources belong to two different clusters. We thus have a very large number of clusters, N, corresponding to a number of all jokes that have nothing in common, at least according to GTVH.

We are no longer looking at a binary problem, but at an N-classification problem[7]. We will further assume that determining a number of N is outside of our problem space, and that at a centre of each cluster, there is a joke that has nothing similar with any other joke cluster centres. Our next question can be summarized as follows: given a joke k, place this joke into clusters i…j where k is similar to the cluster centres i…j, according to the similarity measure. A joke can be similar to more than one cluster. For example, jokes (4) and (5) could form the centre of clusters because they have nothing in common according to GTVH. Joke (6), however, belongs to some degree to both clusters as it is similar to (4) in, at the very least, target, and it is similar to (5) in one of the script overlap/oppositeness pairs. Admittedly, if one followed an all-or-nothing method, (6) belongs to the cluster with joke (5) serving as a

centre. However, if we were interested in all jokes about Irish, not placing (6) into cluster with (4) serving as a centre would potentially lose this joke.

This brings us to the next problem of classification: what is similar? From the GTVH we know that two jokes that share three knowledge resources are more similar than two jokes that share two knowledge resources. Two jokes that share one knowledge resources that is higher in the hierarchy is more similar than two jokes that share one knowledge resources that is lower in the hierarchy. A question to be raised is, what does it mean to share a knowledge resource?

Davies (1990) provides a table of over 30 counties where stupidity jokes are told. The targets of these jokes correspond to people that are butts of the stupidity jokes in a corresponding county. For example, in Britain the targets are Irish, in New Zealand the targets are Irish and Maori, in India—Sikhs, in Austria—Burgenlanders and Carinthians, in Nigeria—Hausas. Clearly, a Target knowledge resource is somewhat different in all of these jokes, but are they all different to the same degree? Is the difference between Sikhs and Irish the same as the difference between Burgenlanders and Carinthians? Is the difference between Irish and Maori (both New Zealand's targets) more than the difference between Burgenlanders and Carinthians (both Austria's targets)? Where do Kerrymen, the target of Irish jokes (Davies 2008), belong? An elegant explanation, and a possible solution to the dilemma, is Davies' explanation of the relationship between the targets and the counties where jokes are told: "[targets] live on the geographical, economic, cultural or linguistic periphery of the peoples in the first column [the country]." And, thus, for this case, Maori and Irish should be closer than Carinthians and Irish because their pair-wise difference or similarity is derived relative to where the jokes are told.

We will assume for the moment that we know the source of a joke—not a trivial task for internet-based humour collection. However, there is a bigger issue here: Maori and Irish should be similar only as far as New Zealand's stupidity jokes are concerned. They may not be close at all in other forms of humour. For example, replacing Irishman with Maori in joke (4) is unlikely to be as successful as the original. A non-bona-fide comparison between these groups will give very different similarity values, mostly because it won't be relative to a geographic region where the jokes are common, it is likely to be relative to other information typically assumed about people. It is the relative clustering of jokes with the number of clusters corresponding to the number of jokes with non-overlapping features, based on at least four GTVH knowledge resources (SO, NS, SI, LA), that could provide similarity metric for target comparison.

From a modelling perspective, Davies' questions of "What are the common factors that characterize the relationship between each pair of jokers and persons joked about?" can be rephrased as "what factors characterize relationships between each of the four knowledge recourses, together or separate, and the target?" From the implementation perspective—which is concerned with what is needed to build a working computational humour system—this knowledge is needed to be able to adjust targets to known jokes in order to deliver context appropriate humour in human-computer communication.

Where do non-jokes belong?

Assuming that we have a set of jokes that is nicely clustered according to the previous section, one may start thinking whether it is necessary to add non-jokes into the mix. Non-bona-fide text should have at least one pair of scripts that overlap and oppose. Bone-fide text, hopefully, do not have such pairs (unless it was unintentionally added). This means that there are at least three more GTVH knowledge resources to compare, Language (LA), Narrative

Structure (NS), and Situation (SI), that may be similar enough with a joke. As an example, consider (7):

(7) Cop pulls over Google self-driving car, finds no driver to ticket.[8]

First of all, let us assume that even if this sentence is amusing, this is not enough to be a joke: there is no salient pair of scripts that overlap and oppose at the same time, thanks to various technological advances. Second, while there is no drunken Irishman behind the wheel, the situation is somewhat similar: a cop stops a car due to some violations. If we were to include bona-fide texts in our clusters, this text would fit into the cluster with joke (4) serving as the centre. The question is, what can we learn from the addition of bona-fide texts into the corpus. On the one hand, even if we don't learn anything, we are not being penalized as we do not need an extra dimension: no salient script pair implies bona-fine communication, and bona-fide communication implies no salient script pairs resulting in overlap and oppositeness. On the other hand, we have information about a similar situation that should follow typical scripts, with every atypical part being described and explained.

This can serve as an excellent source of knowledge (for a computer) about what is approximately normal (something has to be abnormal if it hits the news). On the other hand, capturing bona-fide information that is as similar as possible to a given joke cluster is a very time-consuming task that centres on, again, similarity. At what point do we consider a text similar enough for it to be useful and included? Should similarity be calculated only based on knowledge resources? Should length of text be included in the calculation of similarity? What is the minimum value below which text would be discarded? Is it enough to be similar to one cluster, or should a number of clusters with above threshold calculation be included? These are the questions that are typically not considered when a human-centric approach is used: after all, you cannot unknown something.

Non-jokes can also serve as a source of predictions of new jokes. A new bona-fide event that is reported and that is similar to a previous bona-fide event that generated jokes, may generate jokes itself. This could serve as a triggering mechanism to look for such new jokes as well as an opportunity for a computer to adjust previous jokes to the new event.

Finally, if a widely reported new bona-fide event does not generate jokes, as predicted by similar previous bona-fide events, one could start looking for a cause of such absence. This again, is something that is learned from Davies (2008): "the search for jokes that could have been invented or co-opted but which do not exist."

How much is enough?

In the previous discussion, we ignored computational complexity: with N clusters of joke 'types' which contain M bona-fide event descriptions that are similar at least to one of the clusters, we are adding L new events that are compared first to M, and then to N. However, one may wish to ask a question: at what point is our N large enough; at what point is the number of jokes that are similar with the N centre jokes large enough; at what point is the number of non-joke texts describing M events large enough? We could add the new events into the mix but they will follow the same idea.

It seems obvious that the number of types of jokes that one wishes to analyse depends on the time one is willing to spend on it. It is also obvious that jokes appear all the time and are shared all the time on social media, just like it is possible to go to a library or an archive to get a collection of jokes. Instead of looking at the number of jokes per se, one may wish to look at a time period and, potentially, a geographic region of interest. Such method of collection, while it may not be very effective from a point of view of a discipline that collects

jokes, is a method that a person in natural language processing field will find satisfactory. The task then is to select a time frame that one would be interested in and collect jokes within that timeframe. The hope is that this fishing expedition of, say, k jokes, results in several hundred somewhat dissimilar jokes. Our next task is to separate these jokes in such a way that we have N jokes that have nothing in common with each other (no overlapping knowledge resources). It is at this point that we have k-N jokes that should find their place within N clusters.

We can treat the resulting collection as our base set, and now it is time to go hunting. The question to address here is when to stop hunting—at what point we assume that we have had enough jokes in whatever we wish to learn from a corpus. One answer is that the maximum number of jokes of interest is N||KR||—one joke that differs from others in one and only one KR (depending on whether LM is included or not, it would be five or six). However, it is rather unlikely that all such jokes can be found and we will spend much more time looking for them than for an actual analysis of a corpus. Another way is to analyse the jokes found on a hunting trip, one by one, with respect to their difference (and similarity) to the base joke set. There are multiple measures that calculate distance between two distributions: natural language processing field borrowed a number of such measures from information theory—including entropy, cross entropy, mutual entropy, information gain, and a number of others—and invented new measures, such as various variants of Borrow's Delta (Argamon 2008). A new text can be compared to a collection of existing texts, based on a number of features, and calculate the distance using any of the measures above. We can look at a distribution of these differences through time, as new jokes are hunted down, and stop hunting when the distance measure asymptotically approaches zero. It should be noted that it does not mean that all variants are covered, it just means that we have reached the point of very little gain. This is the time when a footprint of jokes that have not been found could be examined.

Conclusion

Joke comparison is a multidisciplinary task where each discipline—that addresses human aspects—brings a very unique contribution to it. "Jokes are short, compact units which in most cases can be quickly understood and enjoyed […]. [T]hey are invented by, improved by and circulated among large aggregates and networks of individuals. Jokes are a true spontaneous product of the imaginations of and a good reflection of the tastes of ordinary people" (Davies 2008). In order to study such a complex phenomenon and compare and find jokes that could be representative, one should follow well-defined and tested procedures. Christie Davies provides such procedures and explains how to use them.

With the latest advances in Artificial Intelligence, researchers interested in computational humour can (and should!) start looking that joke analysis not only from a 'collection of words' point of view, but from how can such analysis improve theoretical foundations and applications of computational humour. Computational humour papers sometimes look at the linguistics of humour, as it is a fruitful step in getting some results in text-based humour. However, they very rarely look at anything outside of it. Christie Davies' work, while barely known in computational humour circles, can be very useful, not only for a construction of a corpus, but also for its analysis.

Notes

[1] According to GTVH, each joke can be described or represented in terms of Script Overlap/Oppositeness (SO), Logical Mechanism (LM), Situation (SI), Narrative Strategy (NS), Target (TA), and Language (LA), collectively referred to as Knowledge Resources. It should be mentioned that Davies (2004) argued to discard LM as "it is a variable does nothing for the theory."

[2] http://www.allaboutparkinsons.com/forum/archive/index.php/t-863.html

[3] https://www.cartalk.com/content/lame-jokes-8

[4] https://www.startasl.com/deaf-jokes_html

[5] https://www.yelp.com/topic/los-angeles-irish-joke-gone-deaf

[6] In (computational) linguistics, a semantic role means a semantic relationship between a verb in a sentence and noun phrases that are governed by the verb. For example, in a sentence *A rock broke the vase*, a rock is an instrument semantic role, the vase is a patient semantic role.

[7] For the purposes of this chapter, we do not differentiate between clustering and classification as in typical Machine Learning tasks. It can be argued here that each non-overlapping joke (at the center of a cluster) can serve as a class.

[8] http://www.cnn.com/2015/11/13/us/google-self-driving-car-pulled-over/index.html.

References

Argamon, S. (2008). 'Interpreting Burrows' delta: Geometric and probabilistic foundations', *Literary and Linguistic Computing*, 23(2), pp. 131–47.

Attardo, S., & Raskin, V. (1991). 'Script Theory revis(it)ed: Joke similarity and the Joke Representation Model', *Humor: The International Journal of Humor Research* 4(3/4), pp. 293–347.

Davies, C. (1990). *Ethnic Humor around the World: A Comparative Analysis*. Bloomington: Indiana University Press.

Davies, C. (2004). 'Victor Raskin on jokes', *Humor: The International Journal of Humor Research* 17(4), pp. 373–380.

Davies, C. (2008). 'Undertaking the comparative study of humor', in Raskin, V. (ed.), *Primer of Humor Research*, Mouton de Gruyter: Berlin/New York, pp. 157–182.

Hempelmann, C. F. (2008). 'Computational humor: Going beyond the pun', in Raskin, V. (ed.), *The Primer of Humor Research*, Berlin-New York: Mouton de Gruyter, pp. 333–360.

Hobbs, J. R. (2009). 'Word meaning and world knowledge', in Maienborn, C., von Heusinger, K., Portner, P. & van Leusen, N. (eds.), *Semantics: An International Handbook of Natural Language Meaning*, The Hague: Mouton de Gruyter, pp. 740–761..

Miller, T., Hempelmann, C. F., & Gurevych, I. (2017). 'SemEval 2017 Task 7: Detection and interpretation of English puns', Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pp. 56–68.

Raskin, V. (1985). *Semantic Mechanisms of Humor*. Dordrecht: D. Reidel.

Ritchie, G. (2001). 'Current directions in computational humour', *Artificial Intelligence Review* 16 (2), pp. 119–135.

Taylor, J. M. (2017). 'Computational treatments of humor', in Attardo, S. (ed.), *The Routledge Handbook of Language and Humor*, New York, NY: Routledge.