Literature Review

In this section, we review the effectiveness of GAN in text generation and a proposed model, named MaskGAN, which tackles some of the issues that GANs typically see when reviewing text vs. images (William et. al, 2018). It is noted at the beginning, that the most well known training model for text generation are recurrent neural networks. GANs, which are generative adversarial networks, tackle training by having a generator and discriminator. The generator creates artificial information based on being trained through a training set, and the discriminator looks to discern what has been artificially generated and what is natural. The discriminator then uses this error and passes it back to the generator, which then attempts to improve its artificial creations, thus causing a looping method that improves both the generator and discriminator.

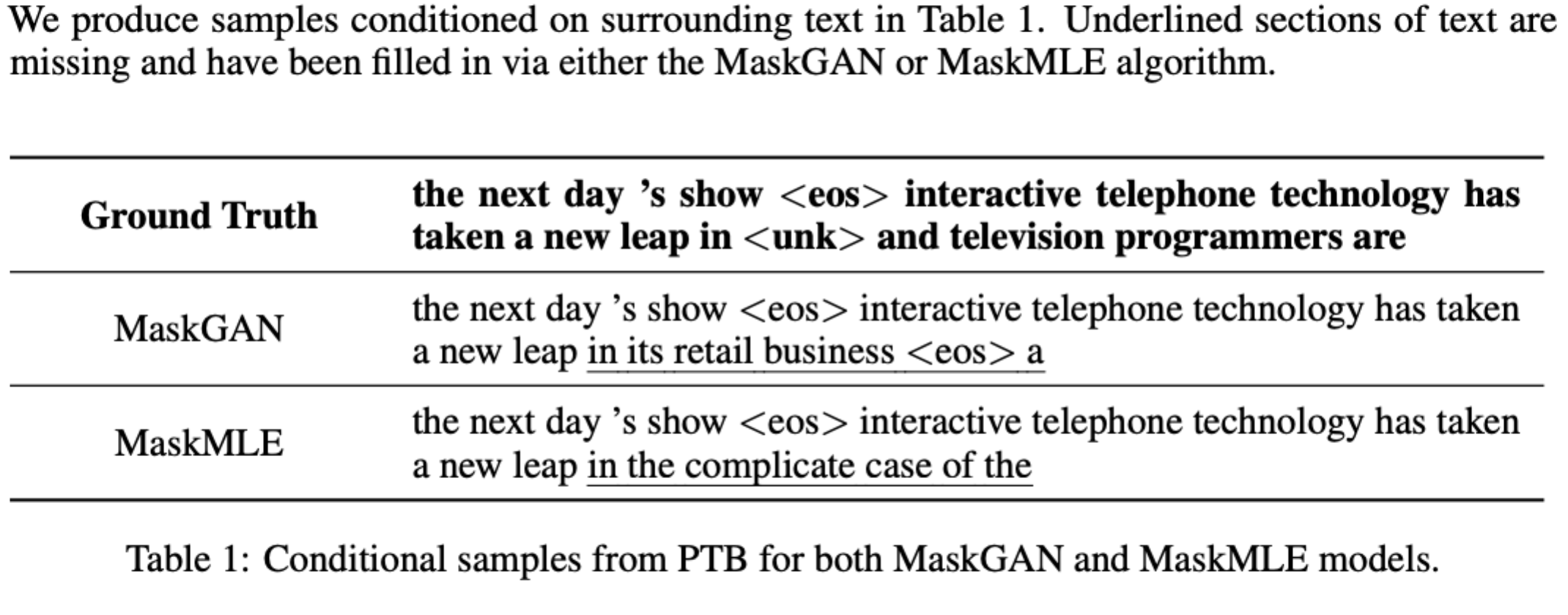
GANs find issue in text generation for a few reasons. First, text is more discrete than continuous, which causes issues when looking to propagate errors throughout (William, et. al, 2018). Additionally, GANs suffer from training instability and mode dropping in relation to text, whereby the instability occurs when the generator finds itself in a non-trained area, or when it receives the reward/penalty only after the entire sequence has been generated rather than by each character/word. To alleviate these issues, the generator and discriminator are trained differently. The generator is trained via reinforcement learning, while the discriminator is trained via maximum likelihood and gradient descent. Additionally, the generator is trained via filling in the blank. The technique from William’s team has been adapted by Bowman et.al (2016), where there are certain redacted sections of sentences that are then generated as tokens and compared to the actual context of the sequences. Through this, the generator can receive more input through reinforcement learning when observing each individual generated token, rather than looking at the entire sequence. This helps reduce the problem of tackling an entire sequence, having the majority of it correct, but because of a single error, the sequence is off skew and causes a major penalty. Thus, the error is more acute in determining which token is the root cause of lowered rewards.

William et.al (2018) describes the architecture of MaskGAN as well as the methodology for training. First, we look toward the architecture. MaskGAN utilizes an encoder/decoder methodology in a seq2seq architecture. MaskGAN approaches similarity to language-modeling by filling in sequences as a whole, but differentiates itself in that it notices its own generated text as well as what it has filled in for itself inside the sequence. The discriminator is also trained via seq2seq, but is given the actual context of the sequence that needs to be filled so that it can make a more accurate analysis on the generated tokens. Williams provides a good example of why this is important. “For instance, without this context, if the discriminator is given the filled-in sequence *the director director guided the series*, it will fail to reliably identify the *director director* bigram as fake text, despite this bigram potentially never appearing in the training corpus (aside from an errant typo). The reason is that it is ambiguous which of the two occurrences of director is fake; the \*associate\* director guided the series or the director \*expertly\* guided the series are both potentially valid sequences. Without the context of which words are real, the discriminator was found to assign equal probability to both words. The result, of course, is an inaccurate learning signal to the generator which will not be correctly penalized for producing these bigrams.”

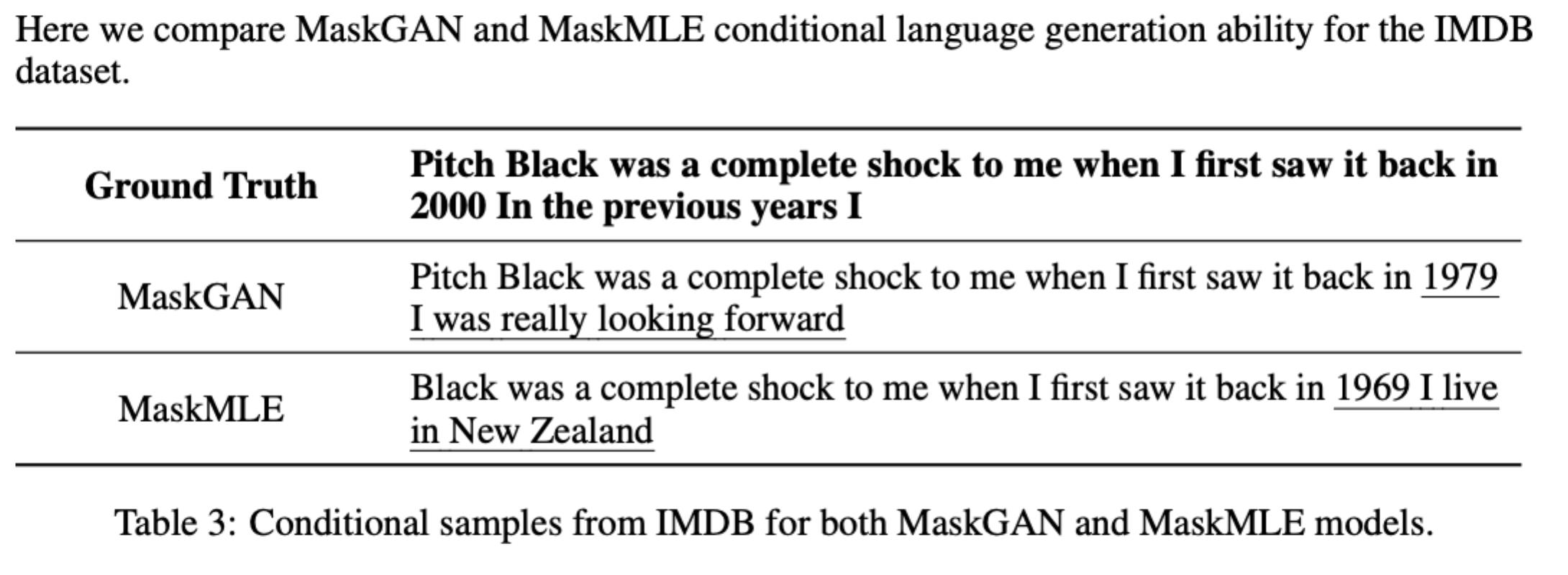
For training, MaskGAN utilizes reinforcement learning as a reward system for teaching the generator how to create new text. The way the reward system works is that when looking at penalties, the penalty will influence the current step as well as every subsequent step for the remainder of the sequence. The reason that we want to attune for entire sequences is because we do not want the model to simply just choose the best token for one specific fill-in and then pick suboptimal tokens for the remainder of the sequence. This counteracts this probability by increasing total penalty while also individualizing each token’s penalty.

One other issue that has been effectively addressed by Williams and co. is that longer sequences can cause issues with how well text is generated. This is addressed by a method whereby the sequence is compared to a length L, and if it expands beyond length L, it increases it to L+1 up until the completion of the filled in sequences. The evaluation of effectiveness of MaskGAN was checked by BLEU score, which incorporates translations from the original text and how it matched out.

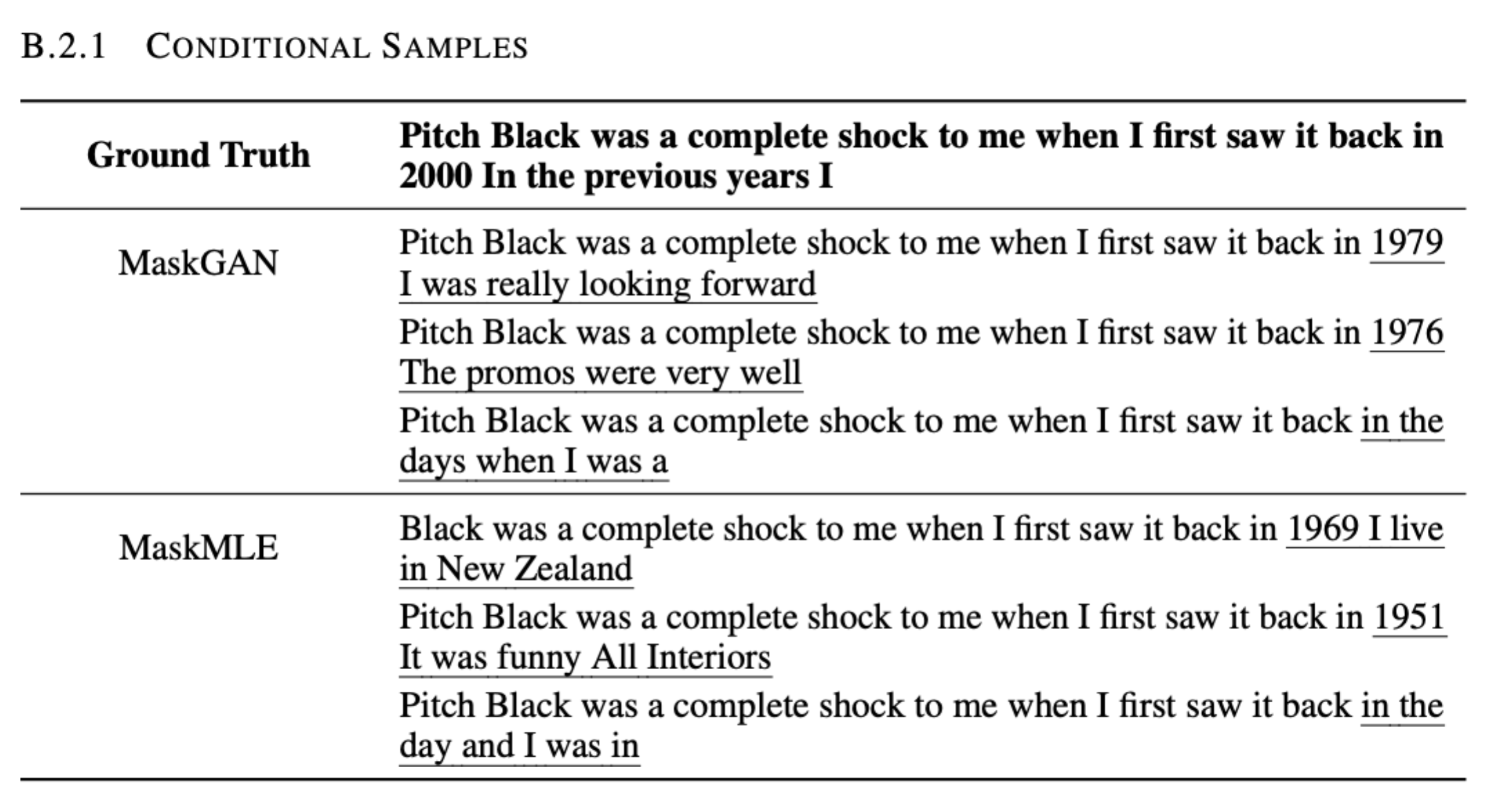
During actual comparisons, MaskGAN was compared to MaskMLE, which is maximum likelihood. The following are excerpts from the article “Better Text Generation Via Filling In The \_\_\_\_,” which shows differences between GAN and MLE for text generation.



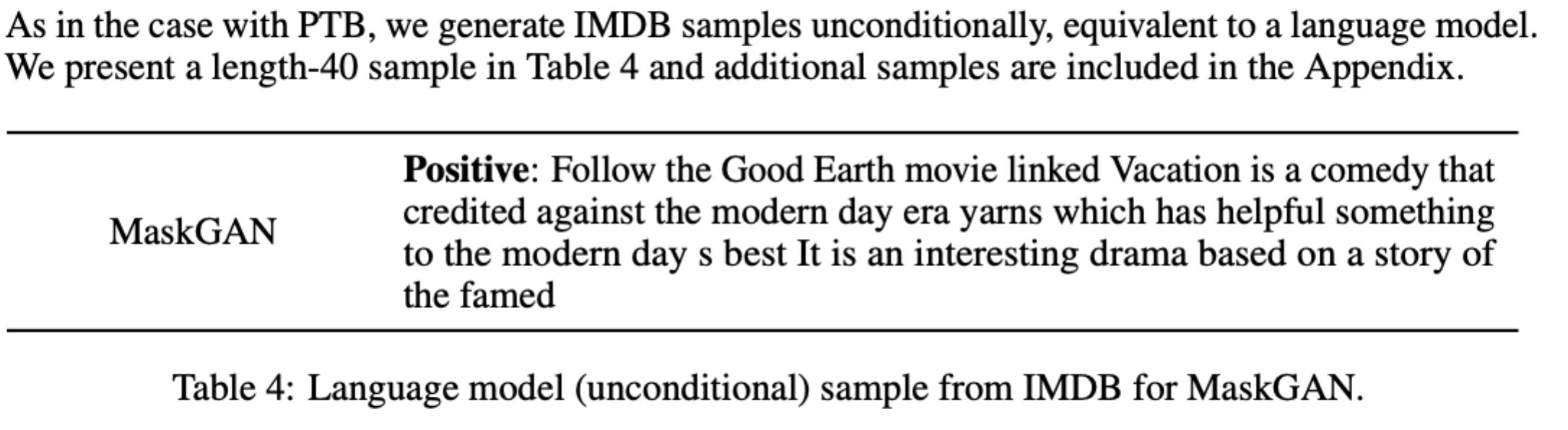
A comparison between MaskGAN and MaskMLE, which shows the improvement of GAN over MLE when generating text via fill in the blank.



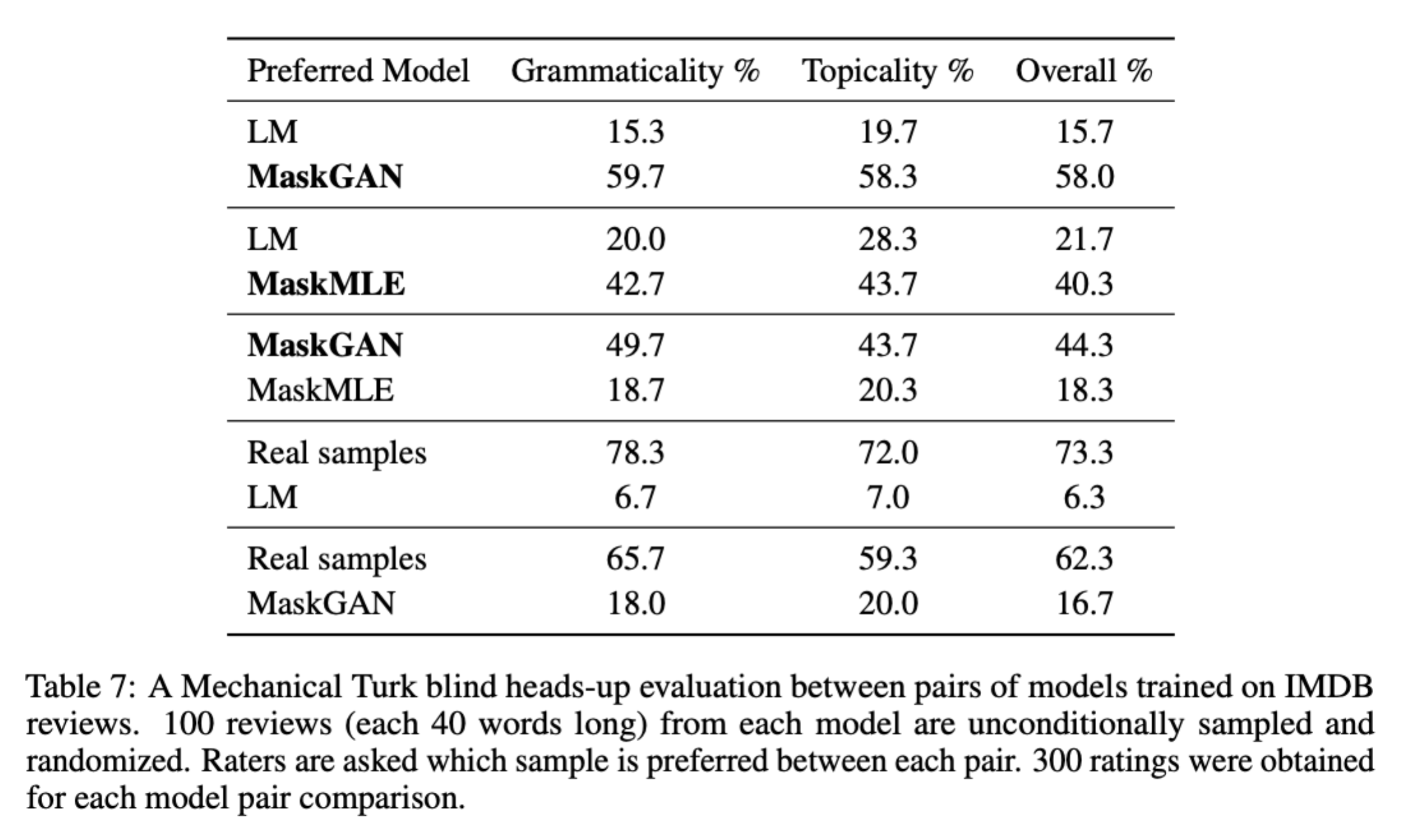
A comparison which takes a look into the IMDB dataset and an example review about Pitch Black. MaskGAN creates a more relevant set of text following the already reviewed text.



Additional comparisons of MaskGAN and MaskMLE using the same IMDB dataset.



MaskGAN generation of a longer sequence text as well as indication of sentiment.



Finally, using evaluators through Amazon’s Mechanical Turk, the effectiveness of MaskGAN vs MaskMLE was compared for true context by humans that can completely understand context and rate appropriately.

In turn, MaskGAN shows a significant improvement over maximum likelihood as the main architecture for text generation. MaskGAN has climbed over major issues that are typical of GANs in the text generation space. Further experimentation involving longer sequences and research into a more contiguous approach vs discrete text approach is expected to produce better results.

William et.al have provided a very comprehensive approach to text generation. While we ourselves are not able to adopt the technique provided in this article, we can appreciate that the proposed idea not only looks at each generated token, but also takes a more holistic approach to the entire sentence from the generator and properly penalizes it during the discriminator step, as this is an issue that occurred during our own investigation into GAN architecture for text generation.

References

William Fedus, Ian Goodfellow, Andrew Dai. MaskGAN: Better Text Generation Via Filling In The \_\_\_\_\_\_\_, 2018, <https://arxiv.org/pdf/1801.07736.pdf>.

Samuel Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In 20th SIGNLL Conference on Computational Natural Language Learning, 2016.