UNIVERSITY OF AMSTERDAM

MSc ARTIFICIAL INTELLIGENCE
MASTER THESIS

A Plug-and-Play Approach to Age-Adaptive Dialogue Generation

---

by

LENNERT JANSEN

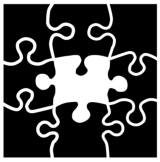
10488952

October 2, 2021

Number of Credits
Period in which the research was carried out

Supervisor:
Dr SANDRO PEZZELLE

Assessor:
Dr RAQUEL FERNÁNDEZ



INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

Contents

Chapter 1

Introduction

In recent years, we have witnessed promising advances in natural language processing (NLP) tasks, such as language modeling, reading comprehension, machine translation, controllable text generation, and conversational response generation [Radford et al., 2019, Bahdanau et al., 2015, Dathathri et al., 2020, Madotto et al., 2020]. Vaswani et al. [2017]'s Transformer architecture plays a central role in many of the state of the art (SotA) solutions to these problems. Transformer-based language models (LMs) pre-trained on massive amounts of textual data, most famously OpenAI's GPT-2 [Radford et al., 2019], have demonstrated their usefulness for several of the aforementioned NLP tasks. For instance, controllable text generation and producing dialogue responses have improved greatly because of GPT-based hybrid models.

[L: CTG comes out of the blue in the following paragraph. Introduce it a little bit by describing what is is, and why/how it is an important task.]

- Controllable text generation entails generating text samples that possess a predefined textual property, like having a positive sentiment, or being about a certain topic.
- Controlling more fine-grained linguistic properties, like resemblance of age-specific vernacular, still poses an important, yet unsolved/insufficiently studied (?) challenge.
- Personalized interaction between humans and AI systems is crucial to obtain systems that can be trusted by users and are perceived as natural.
- (Age-)adaptive language generation can be used to personalize AI-powered personal assistants like Siri and Alexa, improving user experience and trust.

- It is important for AI-power conversational agents to be accessible to varying user profiles, rather than targeted at one particular user group.

- In this work, I/we focus on one aspect that may influence successful personalization of conversational agents: user age profile.

Controllable text generation (CTG) aims to enforce abstract properties, like writing style, on the passages being produced. Fine-tuning large-scale LMs for writing-style adaptation is extremely expensive, but Dathathri et al. [2020] and Li et al. [2020] propose methods that both excel at the task, while bypassing significant retraining costs. Dialogue response generation is the task of producing replies to a conversational agent's prompts, in a manner that is ideally both non-repetitive and relevant to the course of the conversation. With DialoGPT, Zhang et al. [2020] also manage to leverage GPT-2's powerful fluency for dialogue tasks, by framing them as language modeling tasks where multi-turn dialogue sessions are seen as long texts.

[L: Introduce dialogue response generation a bit more. Also emphasize its importance. And then introduce the combined task and its importance.] A blend of CTG and dialogue response generation, i.e., controllable dialogue response generation, is an interesting and only partially explored route. It ties closely to one of Artificial Intelligence's long-standing goals of achieving human-like conversation with machines, as humans are known to adapt their language use to the characteristics of their interlocutor [Gallois and Giles, 2015]. Adaptive dialogue generation is difficult due to the challenge of representing traits, like age, gender, or other persona-labeled traits via language expression [Zheng et al., 2019].

In this thesis, I investigate the problem of controllable dialogue generation, with a focus on adapting responses to users' age. As a preliminary research objective, I aim to study to what extent a classifier can detect age-related linguistic differences in natural language, and which features are most helpful in age-group detection. Do they (i.e., the linguistic or latent features exploited by the classifier) match the age-related informative features reported in previous work? After empirically confirming that speaker age detection is possible, I explore whether large-scale LMs, e.g. GPT-2, can be leveraged for text generation, controlled for age-groups. And what role does the used data play in the differences in output and performance between regular GPT-2 and controllable GPT-2? Finally, my research focuses on the degree to which such a CTG model is successful in generating dialogue that is adaptive w.r.t. age, such that it has a detectable effect on the perception of the user.

The remainder of this thesis is structured as follows: Chapter 2.1 positions the subject of controlled text generation in its theoretical background, and and 2.2 compares it to the most relevant related work. The methodology in Chapter **??** gives detailed explanations of the most important modeling methods and techniques used for this research.

Chapter 2

Literature review

[L: TODO - Add a brief general introduction to the chapter here. See Sandro's comment. E.g:]

We provide a literature review ... to introduce important concepts ... and work related to ours.

2.1 Background

Controllable dialogue generation encompasses several concepts in natural language processing and linguistics that must be understood to approach the problem. This subsection highlights these topics and positions the central problem of this thesis in its relevant theoretical background.

[L: Keep in mind the following distinction between Background and Related Work - The Background section should give an overview of the problem and the components involved: dialogue, language generation, dialogue response generation, age modelling, etc., without focusing on one or the other approach — in Related Work, you describe approaches that have been proposed to tackle each of these components, separately or jointly, and which are related or relevant to your own work for some reason]

2.1.1 Language Models

Generally speaking, language modeling is central to many NLP tasks. A language model (LM) is a probability distribution over words in a sentence or document. Language models are trained to predict the probability of the next word in an sentence, given the preceding sequence of words. The language modeling task is formulated as an unsupervised distribution estimation problem of datapoints $\{\mathbf{x}_1, ..., \mathbf{x}_N\}$ (e.g., documents), each representing sequences (of e.g., symbols

or tokens) of varying lengths $(s_{i,1}, ..., s_{i,n}), i \in \{1, ..., N\}$. Note that N denotes the corpus size, and n the sequence length of datapoint i. To avoid cluttered notation, the subscript i will sometimes be omitted when discussing an arbitrary datapoint. The probability distribution over an observation $\mathbf{x}$ (i.e., the joint probability of an ordered sequence) can then be factorized as the product of its constituent conditionals [Radford et al., 2019]:

$$p(\mathbf{x}) = \prod_{j=1}^{n} p(s_j | s_1, ..., s_{j-1}). \tag{2.1}$$

This formulation allows language models to detect and learn patterns in language. The learned representations of these patterns can then be used for a plethora of applications, such as classification, and text generation. Moreover, this results in a framework for tractable sampling from the unconditional language model $p(\mathbf{x})$. $p(\mathbf{x})$ can therefore be seen as a base generative model that can generate sample sentences [Dathathri et al., 2020].

In recent years, the attention-based models, Transformers [Vaswani et al., 2017], have replaced recurrent neural networks (RNNs) as the dominant architecture for LMs, with major improvements in distribution estimation, long-range dependency handling, sample diversity, and parallel processing. Another recent development in language modeling is that of pre-training LMs on massive corpora. So-called large-scale general purpose LMs have demonstrated significant improvements in downstream tasks, i.e., other NLP tasks for which the model was not specifically trained or fine-tuned. Most famously the OpenAI's series of Generative Pre-trained Transformer (GPT) models have improved numerous NLP benchmarks [Radford et al., 2018, 2019, Brown et al., 2020].

2.1.2 (Controllable) Text Generation

In text generation, a language model $p(\mathbf{x})$ is asked to produce text $\mathbf{x}$ given a prompt by sampling from the distribution of words that are assigned the highest likelihood of following the primer text. Text generation in itself is the task of generating a piece of text given an input text. This process can be seen as sampling from a conditional distribution. Controllable text generation refers to the more restrictive problem of enforcing higher-level linguistic features on the generated text during sampling. This can be seen as a sub-problem of vanilla text generation, because the conditioning factor for the output text is further constrained to also include some predefined textual attribute.

This attribute represents a linguistic characteristic of the text, like sentiment, topic, or writing style.

Controllable text generation or CTG is a more challenging problem than vanilla text generation for a number of reasons. First, defining the desired attribute to be controlled for in a manner that it is intelligible for a machine is a challenge in itself. Second, like many NLP problems, there are not many parallel corpora. In the context of controllable generation, parallel corpora are datasets of target and source texts that only differ with respect to some attribute. Furthermore, the measure of attribute adherence is a very vague and ambiguous concept. Namely, a text can be written in an extremely positive sentiment in multiple formulations, all of which adhere to the positive sentiment. Another important hurdle for controllable text generation, especially when CTG is combined to leverage the linguistic power of large-scale language models, is that the cost of fine-tuning or pre-training a model to control for a linguistic attribute can be very high.

2.1.3 Dialogue

[L: TODO - Add a sub-section here about dialogue (also think of a better title for the sub-section). This is necessary because dialogue is the main focus of your thesis. So before you discuss dialogue response generation (which is the NLP-operationalization of dialogue), formally introduce the concept of dialogue. What is it? What does it look like? How is it different from discourse? Provide some examples like we did in the paper-submission.]

[L: See Sandro's suggestions for this subsection.]

Contents:

- introduce what is dialogue,
- what it looks like,
- how it differs from discourse, etc.
- Some examples would also be useful. (maybe from the BNC?)

2.1.4 Dialogue response generation

Text generation is suitable to tackle tasks such as machine translation, abstractive summarization, and paraphrasing. Dialogue response generation is also a special case of language generation. It can be seen as language generation where the prompt is a turn in a dialogue session. Conversational response generation shares open-domain text generation's overarching objective of

producing grammatically correct fluent text, while remaining relevant to the prompt. However, computational dialogue modeling distinguishes itself from most NLP domains due to the challenges associated with modeling human conversation: informal, noisy, unstructured, and even erroneous real-world responses, possibly competing goals of interlocutors, or an inherently more diverse set of acceptable responses. [L: Comment by SF about the previous sentence: *all these aspects could be introduced/described in this subsection about (human) dialogue*]

[L: Consider moving the following paragraph to the methodology. FROM HERE...]

Despite these differences, conversational response generation can be modeled in similar ways to open-domain text generation. Zeng et al. [2020] suggest to either formulate it in terms of source-target pairs, much like neural machine translation, or as a language modeling objective, where the next token or utterance is conditioned on the dialogue history. To remain close to the training objectives of my baseline models (GPT-2 [Radford et al., 2019] and DialoGPT [Zhang et al., 2020]) I choose to adopt the language modeling formulation for conversation modeling. I.e., concatenate all dialogue turns in a multi-turn dialogue session into a long text: $x_1, ..., x_N$. Denote the source sentence or dialogue history as $S = x_1, ..., x_m$ and the target sentence (ground truth response) as $T = x_{m+1}, ..., x_N$. The conditional probability of dialogue continuation given its history $P(T|S)$ can be written as

$$p(T|S) = \prod_{n=m+1}^{N} p(x_n|x_1, ..., x_{n-1}). \tag{2.2}$$

A multi-turn dialogue session $T_1, ..., T_K$ can be written as $p(T_K, ..., T_2|T_1)$ which is essentially the product of all source-target pairs probabilities $p(T_i|T_1, ..., T_{i-1})$. This formulation also shows that optimising the single objective $p(T_K, ..., T_2|T_1)$ is equivalent to optimising all source-target pair probabilities.

[L: ...TO HERE.]

2.1.5 Controllable dialogue generation

[L: Sandro suggests to move the entire following subsection to the Related Work section. See comment. FROM HERE...]

Endowing a dialogue system with personality traits to generate human-like conversation is a long-standing goal in AI. This objective is difficult to reach because of the challenge of representing personality traits via language expression and the lack of large-scale persona-labeled dialogue

datasets [Zheng et al., 2019]. Assuming an encoder-decoder setup, Zheng et al. [2019] argue that most personalized neural conversation models can be classified as one of two types: implicit and explicit personalisation models. For implicit personalization models, each speaker has its own vector representation, which implicitly captures the speaking style of the speaker in the decoding process. These models enjoy the benefit of having a more granular and realistic representation of speaking style, as opposed to a simple discrete set of traits (as is the case for explicit personalization models). On the other hand, it is unclear how speaker style is captured and should be interpreted, as all the information about a speaker's style is encoded in a real-valued vector. Furthermore, these methods suffer from a data sparsity issue, because each dialogue should be tagged with a speaker identifier and there should be sufficient dialogues from each trait-group to train a reliable trait-adaptive model. [L: This should be in the related work. See comment.] This last drawback is a bigger hurdle for the method proposed by Zheng et al. than it is for mine, as their work deals with personalization for intersections of multiple traits, whereas this thesis focuses on adaptation to a small number of age groups.

For explicit personalization models, the generated responses are conditioned either on a given personal profile, text-described persona, or simply an attribute label. That is, speaker traits are represented as key-value pairs or descriptions about age, gender, etc. This can be seen as conditioning the decoder's output on an attribute a, much like the PPLM setup of Dathathri et al. [2020]. Speakers with same set of personality traits can share attribute representations, so it does not require a speaker-specific representation vector. Such structured character descriptions are more explicit, straight-forward, and interpretable. However, explicit personalization models require manually labeled or crowdsourced datasets for development, making it difficult to scale these models to large-scale dialogue datasets.

[L: ...TO HERE]

2.1.6 Language and age

The relationship between a person's age and use of language is a thoroughly studied subject with a decades long history and inherent challenges [Pennebaker and Stone, 2003, Nguyen et al., 2014, Zheng et al., 2019]. A number factors like community membership (e.g., gender, socioeconomic status, or political affiliation), experimental condition (e.g., emotional versus non-emotional disclosure), mode of disclosure (writing versus talking), and other confounding variables complicate the study of age's relation to language [Nguyen et al., 2011]. The relatively

recent advent of widely available computational resources and vast amounts of textual data made it possible to leverage machine learning methods to help detect patterns in language that eluded conventional sociolinguistic research. Early computational investigations into the connection between a person's age and use of language is typically a combination of qualitative and statistical methods. For instance, using a mix between their proprietary count-based text analysis framework, Linguistic Inquiry and Word Count (LIWC) and sociolinguistic theory, Pennebaker and Stone [2003] study the changes in written and spoken language use with increasing age. They discuss four important areas of a person's character that have been found to change with age: emotional experience and expression, identity and social relationships, time orientation, and cognitive abilities. These four axes and their hypothesized relationships with language use and age can be interpreted in the following ways:

1. *Emotional experience and expression:* This is the relationship between increasing age and linguistically observable manifestations of a person's experienced emotions. In practical terms, this is framed as detectable instances of positive and negative affect in language. This complex relationship between age and emotional expression is characterized by decreased levels of negative affect and slightly non-decreasing levels of positive affect. This is also confirmed by the findings of Schler et al. [2006].

2. *Sense of identity and social relationships:* These terms refer to developmental tends in one's relation to self and others, as expressed in their language, e.g., as references to self (*I, me, my*, and *we, us, our*) or others (*they, them, theirs*). Pennebaker and Stone [2003] report that the *quantity* of social connections decreases and the *quality* of remaining relationships increases with age.

3. *Time orientation:* This relationship describes how people express their perception of and orientation towards time. For instance, this can be indicated by the use of time-related verb tenses. The authors suggest that older individuals tend to be more past-oriented than their younger future-oriented counterparts.

4. *Cognitive abilities:* This refers to markers of cognitive capacity in language. Aging is expected to be associated with less use of cognitively complex words after a certain mid-adulthood peak. Specifically, the relationship between markers of cognitive complexity in natural language (cognitive mechanisms, causal insight, and exclusive words) and age is hypothesized to be curvilinear. And because verbal ability does not decline until very

late in life, markers of verbal ability (e.g., use of big words) are not expected to show changes with age.

Pennebaker and Stone [2003] consider the following variables: positive and negative emotions, first-person singular and first-person plural pronouns, social references, time-related words (past-tense, present-tense, and future-tense verbs), big words ($>$ 6 letters), cognitive mechanisms, causal insight, and exclusive words. Their main findings suggest that increasing with age, people use more positive and fewer negative affect words, use fewer self-references, use more future-tense and fewer past-tense verbs, and exhibit a general pattern of increasing cognitive complexity.

Detectable linguistic differences between age-groups can often be attributed to the use of language fads or references to age-specific popular culture. For instance, Schler et al. [2006] find that the use of slang and neologisms (such as *lol* and *ur*) are strong indicators of youth. Similarly, words like 'facebook', 'instagram', and 'netflix' appear in the most frequently used words by younger participants of conversational data collection efforts, like that of the British National Corpus' spoken component [Love et al., 2017].

More recent studies, like that of Nguyen et al. [2011], Zheng et al. [2019], and Abdallah et al. [2020], frame age prediction from text as traditional machine learning problems, like linear regression, support vector machines, or neural architectures. These modeling approaches tend to reveal that strong indicators of age lie at the syntactic or structural level of language use, as opposed to the more content-based lexical level. Furthermore, this could explain why automatic detection from text of more content-based traits, like topic or sentiment, tend to be easier problems to solve than age prediction from text. To emphasize one such complicating factor, Nguyen et al. [2014] argue that differences in language use are often relation to the speaker's social identity, which could differ from their biological identity. This idea that age prediction from text is more challenging than topic or sentiment prediction could be an indication that controlled language generation for age-differences is also a more nuanced problem than topical steered text generation.

2.2 Related work

[L: Keep in mind the following distinction between Background and Related Work - The Background section should give an overview of the problem and the components involved:

dialogue, language generation, dialogue response generation, age modelling, etc., without focusing on one or the other approach — in Related Work, you describe approaches that have been proposed to tackle each of these components, separately or jointly, and which are related or relevant to your own work for some reason]

2.2.1 Controllable language generation

Previous approaches to controlled language generation require fine-tuning large Transformer-based language models or training conditional generative LMs from scratch. Most notably CTRL [Keskar et al., 2019], which achieves controllable generation by training a generative Transformer for a number of control codes. CTG models that require fine-tuning for control, like CTRL, can produce high quality fluent text because they are specifically trained to maximize the likelihood of generated sequences, given an attribute (denoted $p(\mathbf{x}|a)$), but require training massive language models with computational costs.

Other recent examples of controllable language generation models that are not Transformer-based also exist. Li et al. [2020] introduce OPTIMUS, a large pre-trained Variational Autoencoder (VAE) [Kingma and Welling, 2014] that can be fine-tuned for specific natural language tasks, like guided sentence generation. They demonstrate OPTIMUS' ability to perform controlled text generation from latent style embeddings, with fluency at par with GPT-2. They also show how OPTIMUS generalizes better for low-resource languages than BERT [Devlin et al., 2019]. Nevertheless, much like the previously mentioned CTG models, OPTIMUS still incurs a significant computational cost for fine-tuning per NLP task.

[L: TODO: Where does the following sentence fit best? "The plug-and-play setup of PPLM forms one of the main theoretical foundations of this work."]

The plug-and-play language model (PPLM) [Dathathri et al., 2020] is a recent solution to the problem of high re-training costs of controlled language generation. This approach, inspired by a similar technique for style-control of generated images [Nguyen et al., 2017], leverages the fluency of large-scale language models when controlling them for a specific linguistic attribute, while avoiding incurring significant costs of fine-tuning these massive language models. The main benefit of this setup is its low-cost extensibility. Namely, such large-scale language models are often open-source and available online, and can now be tailored to users' specific needs using a significantly easier to train attribute model. The original architecture proposed by Dathathri et al. uses GPT-2 as a base language model which provides grammatical fluency, combined with

a significantly easier to train attribute model (i.e., a simple BoW or single-layer classifier). Using gradient updates to the activation space of the much smaller attribute model, they manage to generate language that combines (some of) the fluency of GPT-2 with the stylistic control of the attribute model, without the cost of retraining a specialised architecture. They demonstrate that PPLM achieves desirable fluency (i.e., perplexity measured with GPT(-1) [Radford et al., 2018]), as well as measurable attribute control. Their architecture's applicability is also demonstrated on tasks such as controlled story writing and language detoxification. They also show a clear trade-off between attribute control and grammatical correctness and diversity.

2.2.2 Text style transfer

Text style transfer is the task of changing a text's stylistic properties, while retaining its style-independent properties, like content and fluency [Dai et al., 2019]. Text style transfer is a closely related problem to controllable language generation. Its similarity lies in trying to modify the output distribution of a language generation model, such that stylistic characteristics of the produced text are controllable, keeping content and fluency preserved. It involves rewriting an input text with a specific style. More formally, given a text $\mathbf{x}$, its corresponding style-representing vector $\mathbf{s}^{(i)}$, the number of different styles K over which there exists a distribution, and a desired style $\hat{\mathbf{s}} \in \{\mathbf{s}^{(i)}\}_{i=1}^{K}$, the goal of text style transfer is to produce output text $\hat{\mathbf{x}}$ with style $\hat{\mathbf{s}}$, and the style-independent properties of $\mathbf{x}$.

Previous approaches to text style transfer involve passing input text through an RNN-based encoder, yielding a style-dependent latent representation $\mathbf{z}$ [Zhang et al., 2018]. Typically, these approaches then attempt to "disentangle" $\mathbf{z}$ into a style-independent content representation and a latent representation of the stylistic properties of the input text. The subsequent decoder then receives the content representation and a new latent style variable as input, to ultimately produce a style-altered output text with unchanged content. This style-disentanglement approach has a number of drawbacks: **(1)** It is difficult to evaluate the quality of disentanglement of the latent space. **(2)** It is hard to capture rich semantic information in the latent representation due to limited capacity of vector representations (especially for long texts). **(3)** To disentangle style and content in the latent representations, all previous approaches have to assume all input texts can be encoded by a fixed-size latent vector. **(4)** Since most previous approaches use RNN-based encoder-decoder frameworks, they have problems capturing long-range dependencies in the input sentences. Furthermore, disentanglement might be unnecessary, as Lample et al. [2019]

have shown a proper decoder can perform controllable text generation from an entangled latent representation by "overwriting" the original style.

To address these drawbacks, Dai et al. [2019] propose Style Transformer, a Transformer-based alternative encoder-decoder framework for text style transfer. The authors' approach does not require any manipulation (i.e., disentanglement) of the latent space, eliminates the need for a fixed-size vector representation of the input, and handles long-range dependencies better due to Transformers' attention mechanism. Aside from this being the first application of Transformers for text style transfer, Dai et al. [2019] contribute a novel training algorithm for such models, that boasts significant improvements of results on two text style transfer datasets.

2.2.3 Dialogue Generation

Dialogue generation is task of automatically generating a response given a user's prompt. Zhang et al. [2020] introduce DialoGPT, a tunable large-scale language model for generation of conversational responses, trained on Reddit discussion chain data. DialoGPT therefore extends GPT-2 [Radford et al., 2019] to address a more restrictive sub-category of text generation, i.e., conversational response generation. DialoGPT inherits from GPT-2 a 12-to-48 layer transformer with layer normalization, a custom initialization scheme that accounts for model depth, and byte pair encodings [Sennrich et al., 2016] as a tokenizer. The generation task remains framed as language modeling, where a multi-turn dialogue session is modeled as a long text.

To address the well-known problem of open-domain text generation models producing bland and uninformative samples, Zhang et al. [2020] implement a maximum mutual information (MMI) scoring function. MMI uses a pre-trained backward model to predict $p(\texttt{source}|\texttt{target})$: i.e., the source sentences (dialogue history) given the target (responses, dialogue continuation). First, top-K sampling is used to generate a set of hypotheses. Then the probability $p(\texttt{source}|\texttt{hypothesis})$ is used to re-rank all hypotheses. As frequent and repetitive hypotheses can be associated with many possible queries/sources (i.e., a hypothesis that frequently occurs is one that is apparently applicable to many queries), and maximizing backward model likelihood penalizes repetitive hypotheses, MMI yields a lower probability for highly frequent hypotheses, thereby reducing blandness and promoting diversity.

DialoGPT is evaluated on the Dialog System Technology Challenge (DSTC) 7 track, an end-to-end conversational modeling task in which the goal is to generate conversation responses that go beyond chitchat by injecting information that is grounded in external knowledge. The

model achieves state-of-the-art results on both the human and automatic evaluation results, by achieving near human-like responses that are diverse, relevant to the prompt, much like GPT-2 for open-domain language generation. They train 3 models of small (117M), medium (345M), and large (762M) parameter sizes. The medium-sized 345M model achieves the best automatic evaluation results across most metrics, and is used as one of the baselines in later experiments in this thesis. Their Hugging Face PyTorch implementation can be tested here: `https://huggingface.co/microsoft/DialoGPT-medium`.

Dialogue generation is the essential precursor to this thesis' ultimate task of controllable dialogue generation.

[L: The previous sentence feels out of the blue. Consider removing it or think of a way to create a natural flow towards it.]

2.2.4 Controlled Dialogue Generation

Controlled dialogue generation is the task of steering automatically generated conversational responses to possess certain desired stylistic aspects, like sentiment, specific topic, or more abstract writing style characteristics. Zeng et al. [2020] explore the applications of fine-tuning large language models, like GPT, on (Mandarin and English) medical consultation data. The resulting dialogue systems succeed at generating clinically correct and human-like responses to patients' medical questions. Medical dialogue systems like these can help make healthcare services more accessible and aid medical doctors to improve patient care.

Zheng et al. [2019] investigate the problem of incorporating explicit personal characteristics in dialogue generation to deliver personalized conversation. They introduce a dataset `PersonalDialog`, which is a large-scale multi-turn dialogue dataset with personality trait labeling (i.e., `Age`, `Gender`, `Location`, `Interest Tags`, etc.) for a large number of speakers. And Zheng et al. [2019] propose persona-aware models that apply a trait fusion module in the encoder-decoder framework to capture and address personality traits in dialogue generation. Persona-aware attention mechanisms and bias are used to incorporate personality information in the decoding process. All their tested classification and dialogue generation models are either variations of RNNs (such as LSTMs or gated recurrent units (GRUs)), convolutional neural networks (CNNs), or hybrids of these systems (LSTM-outputs fed into a CNN, known as recurrent convolutional neural networks (RCNNs)). The authors study the influence of age, gender, and location on dialogue classification and generation, and use both automatic (perplexity, trait accu-

racy, and generated response diversity measures) and human evaluation. They find dialogues to be distinguishable by gender (about 90.61% test accuracy), then age (78.32% test accuracy), and finally location (62.04% test accuracy). Both automatic and human evaluation of the generated responses show that the best performing models benefit greatly from the persona-aware attention mechanism, possibly making a case to consider more attention-based architectures instead of RNNs.

Although the previously mentioned architectures are able to produce human-like fluent conversational responses, sometimes even leveraging the fluency of large pre-trained LMs, they all suffer from the same computational drawback. They all require massive amounts of computational power to adapt their language styles, because in their cases, guided generation implies fine-tuning (or even retraining) large attribute-specific dialogue datasets. For general controlled language generation, this obstacle is overcome by Dathathri et al. [2020]'s previously mentioned PPLM setup. The conversational analog of this idea, plug-and-play conversational model (PPCM), is proposed by Madotto et al. [2020]. Similar to PPLM, PPCM achieves guided dialogue generation via activation-space perturbations using easy to train attribute models. Due to the computational complexity of PPLM's decoding process, PPLM is unusable as practical conversational system. PPCM solves this problem by using residual adapters [Bapna and Firat, 2019] to tweak the decoding procedure such that it does not require more computational resources. See Section 4.3.3 for a detailed explanation of the mechanisms behind PPLM and PPCM. Madotto et al. [2020] show, using both human and automatic evaluation, that PPCM can balance grammatical fluency and high degrees of attribute-adherence in its generated responses. PPCM uses DialoGPT as its base language model, and is tested for topical or sentimental attributes (i.e., positive, negative, sports, business, or science & tech). Previous work on controllable language generation focuses on content (e.g., topical attributes, or sentiment), rather than more abstract linguistic features, which I hypothesize are more challenging to model and control. The previously mentioned work by Zheng et al. [2019] is a notable exception, as their approach deals with controlling dialogue systems for linguistic features, like age, gender, and geographical region. However, Zheng et al. [2019] still suffers from significant computational costs, because control is achieved by fine-tuning a large system for every specific set of attributes. Furthermore, their proposed architectures are RNN-based, as opposed to my Transformer-based approach. My work therefore aims to extend the applicability of plug-and-play controlled generation to more abstract linguistic

characteristics than those explored by Dathathri et al. [2020] and Madotto et al. [2020], and without the significant fine-tuning cost of Zheng et al. [2019].

[L: TODO

- Ask for Sandro's feedback on this last rephrased paragraph.
- Is it worth mentioning that Zheng et al 2019 deals with Chinese Mandarin dialogue systems, and mine with English?

]

Chapter 3

Experiment 1: Classification

3.1 Introduction

[L: Briefly introduce the problem you seek to solve (i.e., detection of age related linguistic features from dialogue and discourse), your hypotheses, and give an overview of the chapter (i.e., data, methods and models, results, and analyses).]

3.2 Data

We use a dataset of dialogue data where information about the age of the speakers involved in the conversation is available (see the dialogue snippets in Figure), i.e., the spoken partition of the British National Corpus Love et al. [2017]. We henceforth refer to it as our *dialogue* dataset. For comparison with previous work, and to explore commonalities and differences between various types of language data, we also experiment with a dataset of discourse, i.e., the Blog Authorship Corpus used by Schler et al. [2006], that we henceforth refer to as our *discourse* dataset. Below, we briefly describe the two datasets along with the pre-processing steps we took to make the data suitable for our experiments.

dataset	# age groups	# samples	# tokens	mean length (±std)	min - max length	# topics
dialogue	2	64,994	10.4M	6.4 (±9.9)	1 - 724	790
discourse	3	677,244	140M	102.2 (±212.9)	1 - 71,580	40

Table 3.1: Descriptive statistics of the datasets used in our experiments. Length is the number of tokens in a sample.

age 19-29
A: oh that's cool
B: different sights and stuff
A: oh

age 50+
A: well quite and I'd have to come back as well
B: that's of course
A: and make up for you know

Figure 3.1: Example dialogue snippets from speakers of different age groups (19-29 vs. 50+) in the British National Corpus. We conjecture that stylistic and lexical differences between age groups can be detected. In our approach, we experiment at the level of the utterance.

3.3 Dialogue Dataset

This partition of the British National Corpus includes spoken informal open-domain conversations between people that were collected between 2012 and 2016 via crowd-sourcing, and then recorded and transcribed by the creators. Dialogues can be between two or more interlocutors, and are annotated along several dimensions including age and gender together with geographic and social indicators. Speaker ages in the original dataset are categorized in the following ten brackets: 0-10, 11-18, 19-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, and 90-99.

We focus on conversations in the British National Corpus that took place between two interlocutors, and only consider dialogues between people of the same age group. We then focus on dialogues by speakers belonging to two age groups: 19-29 and 50+, in which we group conversations from five original brackets: 50-59, 60-69, 70-79, 80-89, and 90-99. We omit the intermediate age bracket to allow for clearer differentiation.

We split the dialogues into their constituent utterances (e.g., from each dialogue snippet in Figure 3.1 we extract three utterances), and further pre-process them by removing non-alphabetical characters and stopwords, by using the Natural Language Toolkit (NLTK) English stopword list Bird et al. [2009]. Only samples which were not empty after pre-processing were kept. The resulting dialogue dataset, that we use for our experiments, includes around 65K utterances with an average length of 6.4 tokens. Descriptive statistics of it are reported in Table 3.1.

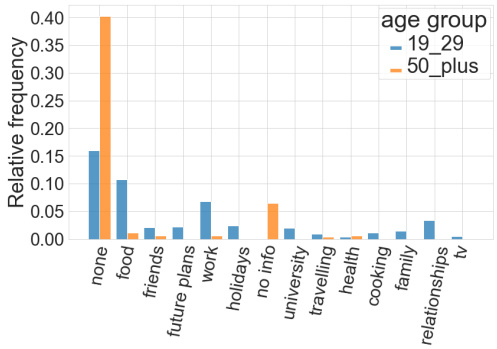
Each conversation in the British National Corpus is annotated with a list of *topics* provided by the speakers during data collection. To extract a single representative topic from this list, we first compute the frequency of all topic labels in the whole dataset. Then, for each utterance, we take the label in the conversation with the highest frequency in the ranking. In total, our final dataset includes 790 unique topic labels. The distribution of the most frequent ones is reported

in Figure 3.2a. As can be seen, frequent topics (besides the frequent *none* label) are *food*, *work*, and *holidays*, which reveals the colloquial and everyday nature of the dialogues in this dataset.

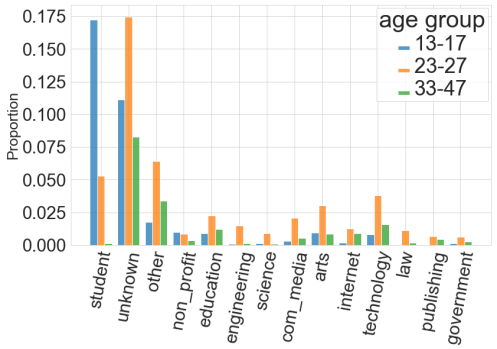
3.4 Discourse Dataset

The Blog Authorship Corpus Schler et al. [2006] is a collection of blog posts posted on `https://www.blogger.com`, gathered in or before August 2004. Each blog entry is written by a single user whose age, gender, and astrological sign are reported. The corpus contains almost 700,000 posts by 19,000 unique bloggers (i.e., ∼35 posts per blogger on average). For our experiments, similar to Schler et al. [2006], we consider three age groups: 13-17, 23-27, and 33+. We pre-process the data in the same way as described above, namely by removing stopwords and non-alphabetical characters. The resulting dataset, that we use for our experiments, includes slightly more than 677K samples with an average length of 102.2 tokens. Descriptive statistics of it are reported in Table 3.1.

Each sample in the Blog Authorship Corpus is annotated with one topic. In our final discourse dataset, the unique topics present are 40. Figure 3.2b reports the distribution of the most frequent ones. As can be noted, frequent topics are *student*, *arts*, and *technology*, which reveals that this and the dialogue dataset are rather different.



(a) Distribution of most frequent topics (including the *none* and *no info* labels) in the dialogue dataset, shown by age group. Best viewed in color.



(b) Distribution of most frequent topics (including the *unknown* label) in the discourse dataset, shown by age group. Best viewed in color.

Figure 3.2

3.5 Methodology and experimental setup

We frame the problem as a N-class classification problem: given a fragment of text X, we seek to predict the age class of its speaker/writer. For the dialogue dataset, $N = 2$, while $N = 3$ for the discourse dataset.

We experiment with various models, that we briefly describe here below. Details on the training and evaluation of models are given at the end of the sub-section.

***n*-gram** Our simplest models are based on n-grams, which have the advantage of being highly interpretable. Each data entry (i.e., a dialogue utterance or blog post) is split into chunks of all possible contiguous sequences of n tokens. The resulting vectorized features are used by a logistic regression model to estimate the odds of a text sample belonging to a certain age group. We experiment with unigram, bigram and trigram models. Note that a bigram model uses unigrams and bigrams, and a trigram model unigrams, bigrams, and trigrams.

LSTM and BiLSTM We use a standard Long Short-Term Memory network [LSTM; Hochreiter and Schmidhuber, 1997] with two layers, embedding size 512, and hidden layer size 1024. Batch-wise padding is applied to variable length sequences. The original model's bidirectional extension, the bidirectional LSTM [BiLSTM; Schuster and Paliwal, 1997], is also used. BiLSTM more thoroughly leverages forward and backward directed information by combining the hidden states from both directions. Padding is similarly applied to this model, and the following optimal architecture is found: embedding size 64, 2 layers, and hidden layer size 512. Both RNN-model are found to perform optimally for a learning rate of 10^{-3}.

BERT We experiment with a Transformer-based model, i.e., Bidirectional Encoder Representations from Transformers [BERT; Devlin et al., 2019] for text classification. BERT is pre-trained to learn deeply bidirectional language representations from massive amounts of unlabeled textual data. We experiment with the base, uncased version of BERT, in two settings: by using its pre-trained frozen embeddings (BERT$_{frozen}$) and by fine-tuning the embeddings on our age classification task (BERT$_{FT}$). The BERT embeddings are followed by a dropout layer with dropout probability 0.1, and a linear layer with input size 768.

Experimental details Both datasets are randomly split into a training (75%), validation (15%), and test (10%) set. Each model with a given configuration of hyperparameters is run 5 times with different random initializations. All models are trained on an NVIDIA TitanRTX GPU.

The n-gram models are trained in a One-vs-Rest (OvR) fashion, and optimized using the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm Liu and Nocedal [1989], with a maximum of 10^6 iterations. The n-gram models are trained until convergence or for the maximum number of iterations.

LSTMs and BERT-based models are optimized using Adam Kingma and Ba [2015], and trained for 10 epochs, with an early stopping patience of 3 epochs. The RNN-based models' embeddings are jointly trained, and optimal hyperparameters (i.e., learning rate, embedding size, hidden layer size, and number of layers) are determined using the validation set and a guided grid-search. BERT$_{FT}$ is fine-tuned on the validation set for 10 epochs, or until the early stopping criterion is met. BERT models have a maximum input length of 512 tokens. Sequences exceeding this length are truncated.

The experiments of this study are divided into two phases: (1) automated age detection from written texts or dialogue transcriptions, and (2) age-adaptive dialogue generation.

3.6 Detecting Age-Related Linguistic Patterns in Dialogue

We first report results on *discourse* to check whether we replicate previous findings. Then, we focus on *dialogue* to answer our research questions. We report accuracy and F_1 for each age group.

3.6.1 Classification performance on discourse

Table 3.2 reports the results. As can be seen, all models are well above the baseline in terms of both accuracy and F_1s. This overall confirms previous evidence Schler et al. [2006] that language features of (written) *discourse* can predict, to some extent, the age group to which the person belongs. At the same time, BERT fine-tuned on the age classification task stands out as our best-performing model by achieving highest accuracy (0.731) and highest F_1 in all age groups. BiLSTM and LSTM rank second (0.720) and third (0.714) in terms of accuracy, respectively, while a somehow more mixed pattern is observed for F_1 scores.

Overall, these results indicate that powerful neural models that are capable of representing the linguistic context have a great advantage on this dataset over simpler n-gram models, which are more than 10 accuracy points behind.

Finally, it should be noted that our best results are slightly lower than those obtained by Schler et al. [2006]. This could be due to two main reasons: First, they experiment with a different (smaller) dataset than ours,[1] which also has a different majority baseline (see Table 3.2). Second, while in our approach all models are trained end-to-end on the task, Schler et al. [2006] use hand-crafted features that are specific to the dataset, (as mentioned in the Introduction), which could constitute an advantage.

3.6.2 Classification performance on dialogue

Table 3.3 reports the results. As can be seen, BERT fine-tuned on the task is again the best-performing model in terms of accuracy (0.710), which confirms the effectiveness of this model in detecting age-related linguistic differences. At the same time, it can be noted that the model based on trigrams is basically on par with it in terms of accuracy (0.709) and well above both LSTM and BiLSTM (0.696 and 0.684, respectively). A similar pattern is shown for F_1 scores, where BERT fine-tuned and the trigram model achieve comparable performance, with LSTMs being overall behind (except for the LSTM on age group 50+).

Overall, our results indicate that predicting the age group to which a speaker belongs, using text-based models, is possible also for *dialogue* data, though the task appears to be somehow more challenging compared to when performed on discourse (note that the improvement with respect to the majority/random baseline is lower in dialogue). At the same time, the different ranking of models observed between discourse and dialogue suggests possibly different strategies used by models to solve the task. In particular, the very good performance of the trigram model in *dialogue* suggests that leveraging 'local' linguistic features captured by n-grams is extremely effective in this setup. This could indicate that differences among various age groups are at the level of local lexical constructions. This deserves further analysis, which we carry out in the next section.

3.7 Age detection analyses

We focus our analysis on dialogue. In particular, we compare the two best-performing models, namely BERT$_{FT}$ and the one using trigrams, and aim to shed light on what cues they use to solve the task. We first analyze how these models perform with respect to utterances of various topics. Secondly, we compare the prediction patterns of the two models, which allows us to

[1] They are left with roughly 511K datapoints after pre-processing, while we experiment with around 677K.

Model	**Accuracy** ↑ better	$\boldsymbol{F}_1^{(13-17)}$ ↑ better	$\boldsymbol{F}_1^{(23-27)}$ ↑ better	$\boldsymbol{F}_1^{(33+)}$ ↑ better
Majority class	0.472	*	0.642	*
Schler et al. [2006]	0.762	0.860	0.748	0.504
unigram	0.603 (0.001)	0.760 (0.003)	0.706 (0.001)	0.491 (0.003)
bigram	0.627 (0.001)	0.788 (0.001)	0.715 (0.001)	0.504 (0.002)
trigram	0.625 (0.002)	0.789 (0.001)	0.716 (0.002)	0.485 (0.003)
LSTM	0.714 (0.005)	0.772 (0.007)	0.740 (0.004)	0.501 (0.006)
BiLSTM	0.720 (0.001)	0.778 (0.005)	0.746 (0.001)	0.486 (0.016)
BERT$_{frozen}$	0.604 (0.001)	0.627 (0.011)	0.666 (0.005)	0.198 (0.018)
BERT$_{FT}$	**0.731** (0.002)	**0.791** (0.003)	**0.752** (0.005)	**0.521** (0.020)

Table 3.2: Discourse dataset. Test set results averaged over 5 random initializations. Format: *average metric (standard error)*. Values in **bold** are the highest in the column; in blue, the second highest. *: F_1 is actually $0/0$.

Model	**Accuracy** ↑ better	$\boldsymbol{F}_1^{(19-29)}$ ↑ better	$\boldsymbol{F}_1^{(50+)}$ ↑ better
Random	0.500	0.500	0.500
unigram	0.702 (0.006)	0.713 (0.006)	0.690 (0.006)
bigram	0.703 (0.006)	0.713 (0.005)	0.693 (0.008)
trigram	0.709 (0.007)	**0.718** (0.007)	0.700 (0.008)
LSTM	0.696 (0.005)	0.689 (0.018)	0.701 (0.016)
BiLSTM	0.684 (0.007)	0.688 (0.018)	0.679 (0.016)
BERT$_{frozen}$	0.673 (0.005)	0.679 (0.013)	0.667 (0.018)
BERT$_{FT}$	**0.710** (0.006)	0.717 (0.007)	**0.703** (0.014)

Table 3.3: Dialogue dataset. Test set results averaged over 5 random initializations. Format: *average metric (standard error)*. Values in **bold** are the highest in the column; in blue, the second highest.

highlight easy and hard examples. Finally, we focus on the trigram model and report the n-grams that turn out to be most informative to distinguish between age groups.

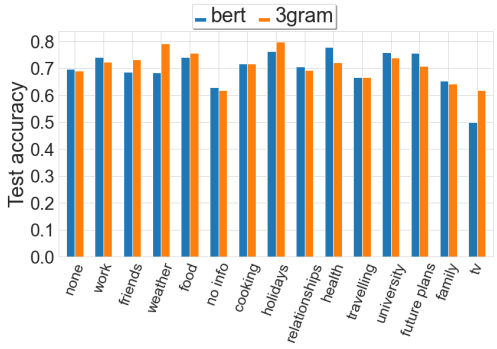
3.7.1 Performance Against Topic

As described in Section 3.3, each utterance in the dialogue dataset is annotated with one label which is representative of its topic.[2] This information is not explicitly available to the models. To explore how the two models deal with utterances in different topical contexts, we compare the accuracy they achieve on the 15 most frequent topics. The results are shown in Figure 3.3a. Two main observations can be made: Firstly, some topics turn out to be generally easier/harder than others, i.e., both models achieve higher/lower performance. To illustrate, both models achieve

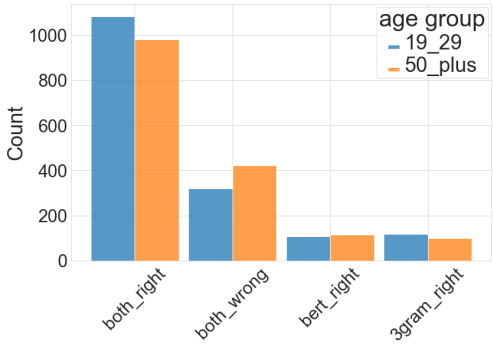
[2]In particular, it represents one of the utterance's topic, i.e., the one most frequently used in the whole data.

age	both correct	both wrong	BERT$_{FT}$ correct | trigram wrong	trigram correct | BERT$_{FT}$ wrong
19-29	oh that's cool	A retrospective exhibition	what even on the green slope?	really?
19-29	a text and then I'll do it	chuck them in those pots	yeah you told me to do you told me to do	and she like won't eat any carbs and she's like
19-29	yeah	mm	somebody made the f***ing table	do you not like total greens?
50+	I said no I don't have them	yeah	really?	my under stairs in the kitchen
50+	that's of course	no no that's alright	it's still we we frequently walk that way	in the first place
50+	oh right	what a tragic life	since this this was new this house?	thank you very much

Table 3.4: Examples where both models are correct/wrong or only BERT$_{FT}$/trigram is correct.



(a) BERT$_{FT}$ and trigram test accuracies per topic for most frequent topics (including none/no info).



(b) Distribution of predicted cases by trigram and BERT$_{FT}$ models for dialogue, split by age groups.

Figure 3.3

an accuracy well above 70% on topics like *food*, *holidays* or *university*, while topics such as *tv*, *family* or *no info* appear to be generally more challenging for both models. While this could be due to (a combination of) various factors, one intuitive possibility is that certain topics allow for more discriminative language features, which could be at the level of the lexicon or the style used to talk about them.

Secondly, some topics appear to be easier for one model rather than the other, and *vice versa*. To illustrate, the trigram model outperforms BERT on the topics *weather*, *holidays* and *tv*, while an opposite pattern is observed for *work*, *health*, and *future plans*. We conjecture that these patterns could be indicative of different strategies and cues exploited by various models to make a prediction. We explore this issue more in-depth in the following section, where we compare the predictions by the two models and qualitatively inspect some examples.

3.7.2 Comparing Model Predictions

We split the data for analysis by whether or not both models make the same correct or incorrect prediction, or whether they differ. Table 3.6 shows the breakdown of these results. As can be seen, a quite large fraction of samples are correctly classified by both models (63.6%), while in 22.9% cases neither of the models make a correct prediction. The remaining cases are almost evenly split between cases where only one of the two is correct. As shown in Figure 3.3b, the

19-29 age group appears to be be slightly easier compared to the 50+ group, where models are observed to make more errors: the trigram misclassifies 50+ utterances 1.26 times as often as 19-29 utterances, and 1.20 times as often by BERT$_{FT}$.

To qualitatively inspect what the utterances falling into these classes look like, in Table 3.4 we show a few cherry-picked cases for each age group. We notice that, not surprisingly, both models have trouble with backchanneling utterances consisting of a single word, such as *yeah*, *mm*, or *really?*, which are used by both age groups. For example, both models seem to consider *yeah* as a 'young' cue, which leads to wrong predictions when *yeah* is used by a speaker in the 50+ group. As for the utterance *really?*, BERT$_{FT}$ assigns it to the 50+ group, while the trigram model makes the opposite prediction. This indicates that certain utterances simply do not contain sufficient distinguishing information, and model predictions that are based on them should therefore not be considered reliable.

This seems to be particularly the case for short utterances. Indeed, through comparing the average length of the utterances incorrectly classified by both models (rightmost column of Table 3.6), we notice that they are much shorter than those belonging to the other cases. This is interesting, and indicates a key challenge in the analysis of dialogue data: on average, shorter utterances contain less signal. On the other hand, short utterances can provide rich conversational signal in dialogue; for example, backchanneling, exclamations, or other acknowledging acts. As a consequence, using length alone as a filter is not an appropriate approach, as it can remove aspects of language use key to differentiating speaker groups.

3.7.3 Most Informative N-grams

Analyzing the most informative n-grams used by the trigram model allows us to qualitatively compare the linguistic differences inherent to each age group. In Table 3.5 we report the top 15 n-grams per group. We find, firstly and intuitively, that colloquial language seems somewhat generational, with unigrams particularly indicative of younger speakers consisting of words such as *cool* and *massive*, and for older speakers, words like *wonderful* and *ordinary*. These unigrams are both informative to the model and indicative of differences in both formality and 'slang' use across age groups.

These most informative n-grams also indicate differences in back-channeling use between age groups; younger speaker's language is more characterized by the use of *hmm*, *um*, *yeah course*, while the top n-grams in the older category will more likely use *yes*, *right*, *right right*. A feature

of younger language also apparent from these examples is in their use of more informal language: *yeah course* rather than *yes*. This informal language use also extends to the use of foul language, which make up a percent of the most informative unigrams shown in Table 3.5.

Interestingly, while topic words make up many of the most informative n-grams for older speakers in Table 3.5, younger speakers are more defined by their use of slang words such as *wanna*, foul language, or adjectives such as *cute*, *cool*, and *massive*. A key finding from Schler et al. [2006] is in the sentiment of language playing an important role, something which some of the most informative n-grams suggest may also be true for the dialogue dataset. As Table 3.5 demonstrates, younger speakers use more dramatic language such as negative foul words, and positive *love, cute, cool*; all words with a strong connotative meaning. This prompts us to hypothesize that further inspection is needed to determine whether the same sentiment pattern will be true of dialogue as it has been reported to be in discourse.

	% cases	**avg. length (±std)***
both correct	63.57%	7.33 (±10.10)
both wrong	22.89%	3.77 (±4.53)
only Trigram correct	6.68 %	6.37 (±6.16)
only BERT correct	6.86 %	6.86 (±7.87)

Table 3.6: Percentage (% cases) of (non-)overlapping (in)correctly predicted cases between trigram and BERT$_{FT}$. *Utterance length measured in tokens.

19-29		50+	
coef.	**n-gram**	**coef.**	**n-gram**
-3.19	um	2.29	yes
-2.91	cool	2.21	wonderful
-2.70	s**t	1.91	building
-2.25	cute	1.86	right right
-2.15	uni	1.80	something like
-2.14	hmm	1.73	garden
-1.97	wanna	1.69	right
-1.93	f**k	1.68	ordinary
-1.91	like	1.67	shed
-1.85	massive	1.63	operation
-1.83	yeah course	1.58	born
-1.81	love	1.57	mother
-1.79	tut	1.55	photographs
-1.74	b***h	1.51	email
-1.68	like oh	1.08	anything like

Table 3.5: For each age group, top 15 most informative n-grams used by the trigram model. **coef.** is the coefficient (and sign) of the corresponding n-gram for the logistic regression model: the higher its absolute value, the higher the utterance's odds to belong to one age group. * indicates masking of foul language.

Notes on the imbalanced BNC.

- Attempts were made to account for the original BNC's bias (i.e., the 19-29 age bracket accounts for roughly 80% of the total considered subset).
- Method 1: weighted loss.
- Method 2: weighted random sampling (i.e., up-sampling of the minority class).
- Weighted random sampling outperformed weighted loss in terms of validation accuracy and F_1 scores, but still failed to surpass the baseline.
- *In terms of test accuracy*, the n-gram models succeeded in beating the baseline (predicting the majority class), whereas the best LSTM and fine-tuned BERT-based failed to do so.
- However, the neural discriminators still outperformed all the other models with respect to minority class F_1 scores, indicating that (1) the n-gram models aren't very useful for correctly classifying the minority class, and that (2) weighted random sampling improved the models' efficacy with respect to the minority class.
- See Appendix A.2 for these results.

Chapter 4

Experiment 2: Generation

4.1 Introduction

[L: Briefly introduce the problem you seek to solve (i.e., plug-and-play adaptive dialogue generation), your hypotheses, and give an overview of the chapter (i.e., data (only BNC, no blogs), methods and models, results, and analyses).]

4.2 Data

[L: Same as Classification data section, but without blog data. What else do I need to say here?]

4.3 Methods for controlled language generation

4.3.1 Transformers

The Transformer architecture plays a central role in most of the recent advances in NLP. The same holds for the methods used in this thesis to investigate controlled dialogue generation and speaker/author age detection. A brief explanation about the Transformer therefore in order. For a more detailed review of the model architecture, the reader is referred to the original paper ([Vaswani et al., 2017]) or this excellent blog post: `https://jalammar.github.io/illustrated-transformer/`.

The Transformer, like most neural sequence processing models, has an encoder-decoder structure. On a high level, the encoder receives an input sequence $\mathbf{x} = (x_1, ..., x_n)$ (e.g., a sentence), and maps this to a sequence of latent continuous variables $\mathbf{z} = (z_1, ..., z_n)$. The decoder then takes $\mathbf{z}$

as input, and maps this to an output sequence $\mathbf{y} = (y_1, ..., y_m)$. Note that the use of positional encodings of the input and output embeddings enables the Transformer to process and generate sequences in arbitrary order, allowing for a high degree of parallelization. The generation of $\mathbf{y}$ happens element-by-element in an auto-regressive fashion, where at step t, element y_{t-1} is also taken as input.

Both the encoder and decoder are comprised of N identical layers (denoted by the 'N $\times$' in the left part of Figure 4.1). Every sub-layer performs a succession of transformations using multi-head self-attention mechanisms and point-wise, fully connected layers, along with residual connections [He et al., 2016] around every sub-layer followed by layer normalization [Ba et al., 2016]. The decoder's first self-attention sub-layer is masked to ensure that the output predictions at sequence position i cannot depend on output positions greater than i. Finally, the decoder passes its output through a linear and softmax layer to produce a probability distribution over the problem space (e.g., the vocabulary) from which the most likely symbols for the generated output sequence $\mathbf{y}$ can be sampled.

A key aspect of the Transformer architecture is its use of attention [Bahdanau et al., 2015]. This allows the encoder-decoder architecture to selectively focus on parts of the input sequence to produce a more informative hidden representation. Vaswani et al. formulate an attention function as a mapping of queries and sets of key-value pairs to an attention output, where matrices represent the queries Q, keys K, and values V. The attention output is a weighted sum of the values, based on the relevance of the corresponding keys to a query. In particular, they employ scaled dot-product attention:

$$\texttt{Attention}(Q, K, V) = \texttt{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{4.1}$$

Furthermore, Vaswani et al. [2017] propose to use multi-head attention by using learned linear projections to project the queries, keys and values h times, and apply the aforementioned attention function to these projections in parallel. The concatenation of these attention outputs, passed through a linear layer, ultimately produces the final output of the Transformer's attention sub-layers. This allows the model to attend to the relevant information from all representation sub-spaces at various sequence positions. See Figure 4.1 for an schematic illustration of the Transformer's structure described above.

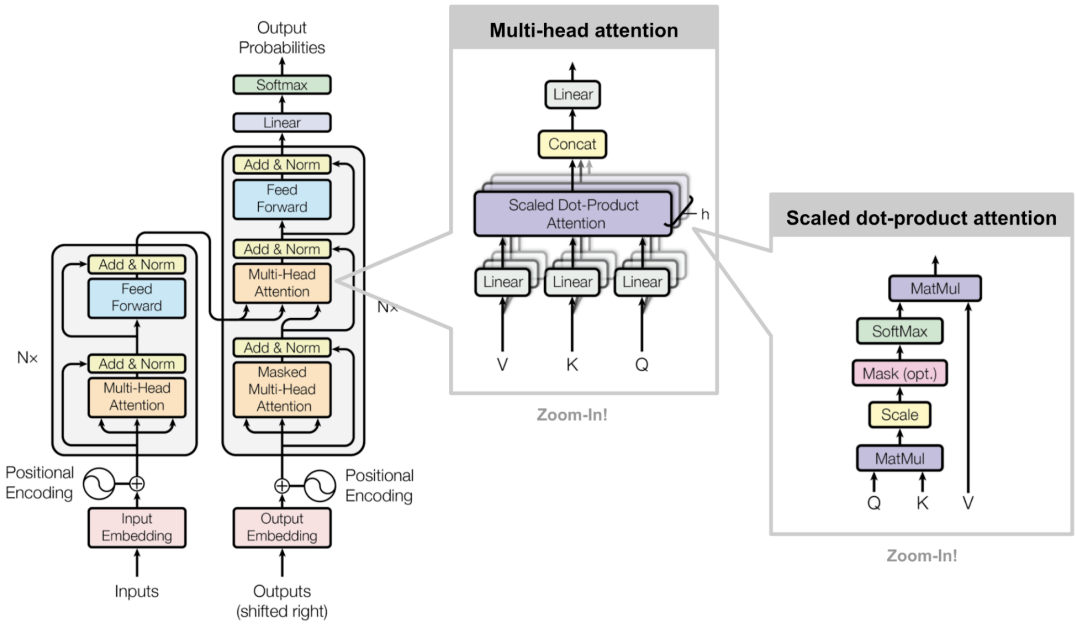


Figure 4.1: An overview of the full Transformer model architecture. *Collated image source:* Fig. 17 in this blog post `https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html`. *Original image source:* Figures 1 and 2 in Vaswani et al. [2017]

4.3.2 Causal language modeling with Transformers

Following the conventions of Dathathri et al. [2020] and Madotto et al. [2020], a dialogue is comprised of multiple alternating turns (sometimes referred to as utterances) between more than one speaker. For simplicity, this project only focuses on dialogues between two speakers. The conversation history at turn t is defined as $\mathcal{D}_t = \{S_1^{(1)}, S_1^{(2)}, ..., S_t^{(1)}\}$, where $S_t^{(j)}$ is speaker j's utterance at time t. Madotto et al. [2020] denote speaker 1 as the user U, and speaker 2 as the conversational system S, yielding dialogue history $\mathcal{D}_t = \{U_1, S_1, ..., U_t\}$. This notational convention will also be used for the user-system experiments later on in this report.

A Transformer-based language model (denoted `LM`) is used in this thesis to model the distribution of dialogues, using dialogue history at time t, $\mathcal{D}_t$, as a prompt to auto-regressively generate the dialogue continuation S_t. More specifically, let the concatenation of the dialogue history at t and its continuation, $\{\mathcal{D}_t, S_t\}$, be represented as a sequence of tokens $\mathbf{x} = \{x_0, ..., x_n\}$. Then, by recursively applying the product rule of probability (Bishop [2006]), the unconditional probability of the sequence $p(\mathbf{x})$ can be expressed as:

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i|x_0, ..., x_{i-1}). \tag{4.2}$$

Dathathri et al. [2020] and Madotto et al. [2020] define the Transformer's decoding process in a recursive fashion. Let H_t denote the conversation history's key-value pairs, i.e., $H_t = \left[(K_t^{(1)}, V_t^{(1)}), ..., (K_t^{(l)}, V_t^{(l)})\right]$, with $(K_t^{(i)}, V_t^{(i)})$ representing the key-value pairs from the `LM`'s i-th layer generated at all time steps 0 through t. This results in the recurrent dedocing process being expressed as:

$$o_{t+1}, H_{t+1} = \texttt{LM}\left(x_t, H_t\right), \tag{4.3}$$

where o_{t+1} is the hidden state of the last layer. Finally, after applying a softmax transformation, the next token x_{t+1} is sampled from the resulting probability distribution, i.e., $x_{t+1} \sim p_{t+1} = \texttt{softmax}\left(W o_{t+1}\right)$, where W is a linear mapping from the model's last hidden state to a vector of vocabulary size. This recursive formulation allows for efficient text generation by leveraging cached memories, without repeated forward passes.

4.3.3 Plug-and-play modeling

Plug-and-play language model (PPLM) Dathathri et al. [2020] works by using a text classifier, referred to as an attribute model, to control the text generated by a language model. Let $p(X)$ denote the distribution of a Transformer-based language model (e.g., GPT-2 or DialoGPT), where X represents the generated text. And $p(a|X)$ denotes the attribute model (e.g., a single-layer or BoW classifier) that represents the degree of adherence of text X to a certain attribute a (e.g., style, sentiment, or age-group characteristics). Then PPLM can be seen as modeling the conditional distribution of generated text X given attribute a, i.e., $p(X|a)$. Note that Bayes' theorem ties these three definitions together as follows:

$$p(X|a) \overset{\text{Bayes' theorem}}{=} \frac{p(X)p(a|X)}{p(a)} \propto p(X)p(a|X). \tag{4.4}$$

To control the generated text, PPLM shifts the aforementioned history H_t (i.e., all Transformer key-value pairs generated up to time t) in the direction of the sum of two gradients:

1. Ascending $\nabla \log p(a|X)$: maximizing the log-likelihood of the desired attribute a under the conditional attribute model. This enforces attribute control.

2. Ascending $\nabla \log p(X)$: maximizing the log-likelihood of the generated language under the original (possibly conversational) language model. This promotes fluency of the generated text.

These two incentive-representing gradients are combined with various coefficients, yielding a set of tunable knobs to steer the generated text in the direction of the desired fluency, attribute control, and length.

Let's first focus on the first of the two gradients, i.e., the attribute control promoting $\nabla \log p(a|X)$. ΔH_t represents the update to history H_t that pushes the distribution of the generated text X in the direction that has a higher likelihood of adhering to desired attribute a. The gradient update rule can be expressed as:

$$\Delta H_t \leftarrow \Delta H_t + \alpha \frac{\nabla_{\Delta H_t} \log p(a|H_t + \Delta H_t)}{\|\nabla_{\Delta H_t} \log p(a|H_t + \Delta H_t)\|^{\gamma}} \tag{4.5}$$

where α is the step size, and γ denotes the normalization term's scaling coefficient. Both step size (α) and the scaling coefficient (γ) influence attribute control. Attribute control can be softened by either decreasing α or increasing γ and vice versa. Note that $\alpha = 0$ recovers the original uncontrolled underlying language model (e.g., GPT-2 or DialoGPT). In practice, ΔH_t is initialized at zero, and the update rule in Equation 4.5 is applied m times (usually 3 to 10), resulting in the updated key-value pair history $\tilde{H}_t = H_t + \Delta H_t$. Then the updated history $\tilde{H}_t$ is passed through the language model, yielding the updated logits (final Transformer-layer): $\tilde{o}_{t+1}, H_t = \texttt{LM}(x_t, \tilde{H}_t)$. And finally the shifted $\tilde{o}_{t+1}$ is linearly mapped through a softmax layer to produce a new, more attribute-adherent, distribution from which to sample, i.e., $x_{t+1} \sim \tilde{p}_{t+1} = \texttt{softmax}\left(W\tilde{o}_{t+1}\right)$.

The method described until now will generate attribute-adherent text, but will likely yield fooling examples [Nguyen et al., 2015] that are gibberish to humans, but get assigned high $p(a|x)$ by the attribute model [Dathathri et al., 2020]. That is why Dathathri et al. [2020] apply two methods to ensure fluency of the generate text. The first is to update ΔH_t such to minimize the Kullback-Leibler (KL) divergence (denoted D_{KL}) between the shifted and original distributions. In practice, D_{KL} is scaled by a coefficient λ_{KL}, typically found to work well for most tasks when set to 0.01. Repetitive text generation (i.e., high $p(a|x)$ but low $p(x)$) can therefore sometimes be avoided by increasing λ_{KL}. The second method to ensure fluency is Post-norm Geometric Mean Fusion [Stahlberg et al., 2018] which, instead of directly influencing ΔH_t like minimizing

D_{KL}, fuses the altered generative distribution $\tilde{p}_{t+1}$ with the unconditional language distribution $p(x)$. This is done during generation by sampling the next token as follows:

$$x_{t+1} \sim \frac{1}{\beta}\left(\tilde{p}_{t+1}^{\gamma_{gm}} p_{t+1}^{1-\gamma_{gm}}\right) \tag{4.6}$$

where β is a normalization constant, p_{t+1} and $\tilde{p}_{t+1}$ denote the original and modified distributions, respectively, and γ_{gn} is a scaling term that interpolates between the two distributions. Because the new sampling distribution in Equation 4.6 converges towards the unconditional language model as $\gamma_{gm} \to 0$, repetitive text generation can be avoided by decreasing the scaling term.

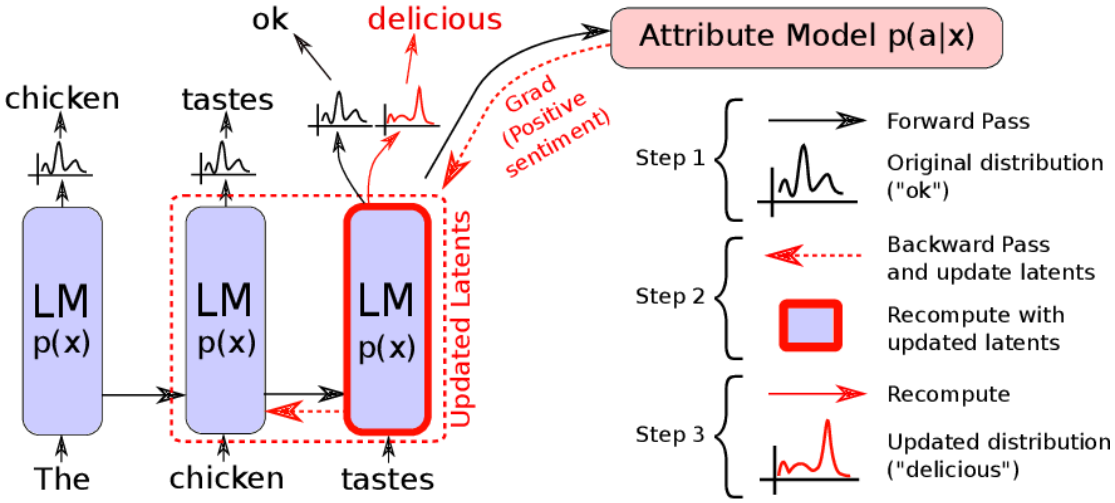


Figure 4.2: A schematic overview of the plug-and-play interaction between attribute model $p(a|\mathbf{x})$ and language model $p(\mathbf{x})$. *Original image source:* Figure 1 of Dathathri et al. [2020]

It is important to realize that the plug-and-play method applied by Dathathri et al. [2020] and Madotto et al. [2020] is different from fine-tuning. Note that in Equation 4.5 the gradient updates are restricted to the history H_t, and do not affect the model's parameters. Because the key-value pairs $(K_t^{(i)}, V_t^{(i)})$ that comprise H_t are activations and not model-weights, the updates only take place in the activation-space. This means that PPLM leaves the underlying (conversational) language model untouched.

Contrary to fine-tuning often massive LMs, PPLM does not incur a significant training cost (depending of course on the complexity of the discriminator or attribute model). However, Madotto et al. [2020] show that PPLM needs a fixed number of m update-steps to for every generated token. This makes the original PPLM setup unsuitable for online interactive applications, like conversational systems. Addressing this problem, they introduce plug-and-play conversational

models (PPCM), which extends PPLM by using the original model setup to generate dialogue datasets with the desired attribute a, and then use optimized residual adapters [Bapna and Firat, 2019] to control `LM`'s output distribution.

Residual adapters are optimizable modules stacked on every Transformer-layer of a pre-trained (language) model. The adapter module then steers the Transformer's output distribution without changing the pre-trained model's weights. A Layer Normalization module [Ba et al., 2016] followed by an auto-encoder with residual a connection constitutes a residual adapter module. More specifically, the residual adapter block can be expressed as the following function composition:

$$\begin{aligned} f_{\theta_i}(x) &= \texttt{ReLU}(\texttt{LayerNorm}(x) \cdot W_i^E) \cdot W_i^D, \\ \texttt{Adapter}(o_{:t}^i) &= f_{\theta_i}(o_{:t}^i) + o_{:t}^i \end{aligned} \tag{4.7}$$

where $o_{:t}^i \in \mathbb{R}^{t \times d}$ denotes the Transformer's i-th layer's latent output at step t, d is the hidden state's size, W_i^E and W_i^D are learnable parameter-matrices of sizes $d \times m$ and $m \times d$, respectively. Finally, m is the auto-encoder's bottle-neck dimension, which is a tunable hyper-parameter for changing the residual adapter's capacity. In practice, Madotto et al. [2020] use PPLM to generate n attribute-adherent dialogue datasets $\mathscr{D}^a = \{\mathcal{D}^1, ..., \mathcal{D}^n\}$, for attribute a. These generated dialogue datasets are then used to train the residual adapter, which they aptly name a plug-and-play adapter, so it can be used to control the language model's output distribution. So for every attribute a, they train the plug-and-play adapter's parameters $\Theta_a := \{\theta_0^a, ..., \theta_l^a\}$, where $\theta_i^a := \{W_i^{E,a}, W_i^{D,a}\}$, such that negative log-likelihood over the corresponding dialogue dataset $\mathscr{D}^a$ is minimized:

$$\Theta_a \text{ s.t. } \min \mathcal{L}(\mathscr{D}^a) = -\sum_{k}^{|\mathscr{D}^a|} \sum_{i}^{n} \log p(s_i^k | s_{<i}^k, \mathcal{D}_t^k), \tag{4.8}$$

where s_i^k is the i-th generated token of response $S_t^k = \{s_0^k, ..., s_n^k\}$ with maximum sequence length n.

4.3.4 Experimental setup and evaluation

Automatic evaluation

Control (attribute-adherence)

How do you measure how representative of the stylistic attribute a the generated text is? Specifically, is the generated text similar to that of the age-group you're controlling for?

Fluency *How do you measure how grammatically correct and fluent the generated texts are?*

- Perplexity
  - **TODO:** When explaining and motivating the use of perplexity as an evaluation metric for (controlled) language models, re-read this piece of documentation about perplexity by Hugging face: `https://huggingface.co/transformers/perplexity.html`
  - $\text{PPL}(\mathbf{x}) = \exp\left\{-\frac{1}{t}\sum_i^t \ln p_\theta(x_i|x_{<i})\right\}$
- Also checkout this blogpost by The Gradient about Evaluation Metrics for Language Modeling (NB: contains BibTeX citation at the bottom): `https://thegradient.pub/understanding-evaluation-metrics-for-language-models/`

Human evaluation

TODO: find humans.

4.4 Results of controlled dialogue generation

Model	ppl. ↓ better	Dist-1 ↑ better	Dist-2 ↑ better	Dist-3 ↑ better	Acc. ↑ better
Baseline	29.60 (±17.54)	**0.89** (±0.09)	**0.93** (±0.07)	0.88 (±0.10)	50.0%
B_{100MCW}	27.80 (±16.42)	0.83 (±0.12)	0.91 (±0.06)	0.88 (±0.09)	50.0%
$B_{Y,FB,80}$	29.13 (±15.17)	0.86 (±0.10)	0.92 (±0.06)	0.88 (±0.11)	70%
$B_{O,FB,80}$	**26.42** (±8.87)	0.86 (±0.10)	0.92 (±0.05)	**0.89** (±0.10)	62.2%
$B_{Y,FB,85}$	28.16 (±14.52)	0.87 (±0.10)	0.92 (±0.06)	0.88 (±0.10)	69.3%
$B_{O,FB,85}$	26.79 (±8.89)	0.88 (±0.09)	0.92 (±0.05)	0.88 (±0.10)	64.2%
$B_{Y,100MIU}$	29.16 (±14.91)	**0.89** (±0.09)	0.92 (±0.06)	0.88 (±0.11)	52.5%
$B_{O,100MIU}$	26.63 (±8.36)	0.87 (±0.10)	0.92 (±0.07)	0.88 (±0.10)	53.7%
$D_{Y,GPT2}$	31.95 (±14.29)	0.82 (±0.17)	0.87 (±0.14)	0.83 (±0.16)	**77.7%**
$D_{O,GPT2}$	33.63 (±24.40)	0.80 (±0.18)	0.87 (±0.11)	0.81 (±0.21)	72.7%
$D^*_{Y,DGPT}$	41.54 (±10.87)	0.91 (±0.11)	0.91 (±0.06)	0.86 (±0.09)	**84.1**%
$D^*_{O,DGPT}$	38.16 (±10.77)	0.87 (±0.11)	0.91 (±0.06)	0.87 (±0.08)	55.6%

Table 4.1: [L: Including stopwords.] Results of age-controlled language generation. Perplexity is perplexity w.r.t. GPT-1. Dist-n is number of distinct n-grams normalized by text length, as a measure of diversity. Acc. is the best BERT model's accuracy when classifying the row's samples.

Notes on Table A.3

- These are initial results.
- All metrics are averaged over 120 samples: 30 samples per group of sequence lengths 8, 16, 32, and 64.
- Young and old accuracy (last two columns) denote the probability of belonging to the young or old age-groups assigned by the best performing BERT-based classifier.
- I use the same parameter settings as Table 6 of Dathathri et al. [2020] to make the results comparable, i.e.:
  - Step-size 0.02. Step-size is α in Equation 4.5.
  - Temperature 1.0.
  - Number of update iterations: 3.
  - γ 1.5.
  - GM-scale 0.9.
  - KL-scale 0.01.
- The current baseline is uncontrolled/unperturbed GPT-2.
- There are four settings for BoW-based control:

  - Young frequency-based wordlist.
  - Old frequency-based wordlist.
  - Young + most informative unigrams as wordlist.
  - Old + most informative unigrams as wordlist.
- Initial observations:
  - Fractions of distinct uni-,bi, and trigrams do not change.
  - Perplexity seems to improve when controlling generation for each age-group, which isn't necessarily what one would expect.
  - The baseline starts off with a higher average probability of belonging to the young age group
  - Controlling for young-language does result in a slightly greater assigned probability of belonging to young age-bracket.
  - Controlling for old-language results in a doubling of the assigned probability of belonging to the old age-bracket.

4.4.1 Bag-of-words control

Frequency-based wordlist generation

Steps taken to create age-specific wordlists (full imbalanced BNC used):

- Remove all stopwords. List of stopwords from NLTK's English stopword list. **TODO: does this make sense? What if differences in use of stopwords are strong indicators of an age-group's speech?**
- Order all unique words by frequency per age-group.
- For both lists, keep the words that account for at least 80% of the respective cumulative probability densities.
- From both sets of words, remove the words that are in the *union* (i.e., the overlapping set) of the young and old sets.
- For both sets, order the words by frequency.
- For both remaining lists, keep the words that account for at least 80% of the respective cumulative probability densities.
- **TODO:** remove curse-words?

- Resulting wordlist lengths:
  - Young (19-29): 90 words
  - Old (50 plus): 225 words

Most informative unigrams as wordlists

4.4.2 Discriminator-based control

Notes on the experimental details of Table 4.1:

- All sequences are generated unconditionally. I.e., from `|<endoftext>|` token.
- All results are averaged over 240 samples.
-

4.5 Controlled text generation analyses

4.5.1 Quantitative analyses

[L: TODO - Add examples of generated sequences along with their model's configurations, age-group, etc. Similar to dialogue snippets earlier.]

Quantitative 1: the effects of generated sequence length

- *Main question: how is generated sequence length related to fluency and control?*
- *Study the relationship between generated sequence length (measured in number of tokens) and automated evaluation metrics (i.e., perplexity, dist-n, and accuracy).*
- *For every metric and for (all?) models, plot sequence length on the x-axis, and the average metric with confidence intervals on the y-axis.*
- *Which patterns do you observe?*

Quantitative 2: the effects of PPLM-parameters on fluency and control

- *Plot and examine the relationship between fluency and control, and various PPLM-parameters (step-size, number of iterations, temperature, top k, gamma, KL-scale).*
- *Which patterns do you observe?*
- [L: How much sense does it make to study this, though? Is that the purpose of my thesis? Hasn't this been studied enough in the PPLM-paper? Which parameters do I choose?]

Quantitative 3: BERT-classifier visualizations.

- ***NB:*** *This is more relevant to the classification experiments, than to the controlled generation experiments.*
- *Use BertViz to visualize what parts of sequences BERT's transformer heads and neurons are focusing on.*
- `https://github.com/jessevig/bertviz`

4.5.2 Qualitative analyses

Qualitative 1: summary statistics and qualitative inspection of various cases

- *Similar to (error)case analyses of Experiment 1.*
- *Provide summary statistics and (**qualitative**) inspection of generated sequences per case.*
- *Cases could be: (1) sequences with low, average, high, or very-high perplexity. (2) (in)correctly classified generated sequences.*
- *What patterns do you observe among, e.g., misclassified sequences with low perplexity?*
- *Provide table of examples per case and age-group. Similar to table 3.4*

Qualitative 2: Human evaluation of fluency, grammaticality, and relevancy

- *Generate and sample text passages for a variety of model-configurations and age-groups.*
- *Have a group of human participants rate these sequences on a scale from 1 to 5 for their (1) fluency, (2) grammaticality, (3) relevance to the prompt (if there is one)*
- *Average the ratings, and compare the human evaluation metrics to the automated evaluation metrics reported in Table 4.1*

Chapter 5

Discussion

...

Chapter 6

Conclusion

...

Bibliography

E. E. Abdallah, J. R. Alzghoul, and M. Alzghool. Age and gender prediction in open domain text. *Procedia Computer Science*, 170:563–570, 2020.

L. J. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL `http://arxiv.org/abs/1607.06450`.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1409.0473`.

A. Bapna and O. Firat. Simple, scalable adaptation for neural machine translation. 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 1538–1548, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1165. URL `https://www.aclweb.org/anthology/D19-1165`.

S. Bird, E. Klein, and E. Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

N. Dai, J. Liang, X. Qiu, and X. Huang. Style transformer: Unpaired text style transfer without disentangled latent representation. *arXiv preprint arXiv:1905.05621*, 2019.

S. Dathathri, A. Madotto, J. Lan, J. Hung, E. Frank, P. Molino, J. Yosinski, and R. Liu. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=H1edEyBKDS`.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

C. Gallois and H. Giles. Communication accommodation theory. *The international encyclopedia of language and social interaction*, pages 1–18, 2015.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

N. S. Keskar, B. McCann, L. Varshney, C. Xiong, and R. Socher. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*, 2019.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. URL http://arxiv.org/abs/1412.6980.

D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

G. Lample, S. Subramanian, E. Smith, L. Denoyer, M. Ranzato, and Y.-L. Boureau. Multiple-attribute text rewriting. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1g2NhC5KQ.

C. Li, X. Gao, Y. Li, B. Peng, X. Li, Y. Zhang, and J. Gao. Optimus: Organizing sentences via pre-trained modeling of a latent space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4678–4699, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.378. URL https://www.aclweb.org/anthology/2020.emnlp-main.378.

D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

R. Love, C. Dembry, A. Hardie, V. Brezina, and T. McEnery. The spoken bnc2014: designing and building a spoken corpus of everyday conversations. *In International Journal of Corpus Linguistics*, 22(3):319–344, 2017.

A. Madotto, E. Ishii, Z. Lin, S. Dathathri, and P. Fung. Plug-and-play conversational models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2422–2433, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.219. URL https://www.aclweb.org/anthology/2020.findings-emnlp.219.

A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 2015.

A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski. Plug & play generative networks: Conditional iterative generation of images in latent space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2017.

D. Nguyen, N. A. Smith, and C. P. Rosé. Author age prediction from text using linear regression. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 115–123, Portland, OR, USA, June 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W11-1515.

D. Nguyen, D. Trieschnigg, A. S. Doğruöz, R. Gravel, M. Theune, T. Meder, and F. De Jong. Why gender and age prediction from tweets is hard: Lessons from a crowdsourcing experiment. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1950–1961, 2014.

J. W. Pennebaker and L. D. Stone. Words of wisdom: Language use over the life span. *Journal of Personality and Social Psychology*, 85(2):291–301, 2003. URL https://doi.org/10.1037/0022-3514.85.2.291.

A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. 2018.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

J. Schler, M. Koppel, S. Argamon, and J. W. Pennebaker. Effects of age and gender on blogging. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, volume 6, pages 199–205, 2006.

M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

R. Sennrich, B. Haddow, and A. Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://www.aclweb.org/anthology/P16-1162.

F. Stahlberg, J. Cross, and V. Stoyanov. Simple fusion: Return of the language model. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 204–211, Brussels, Belgium, Oct. 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6321. URL https://www.aclweb.org/anthology/W18-6321.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

G. Zeng, W. Yang, Z. Ju, Y. Yang, S. Wang, R. Zhang, M. Zhou, J. Zeng, X. Dong, R. Zhang, H. Fang, P. Zhu, S. Chen, and P. Xie. MedDialog: Large-scale medical dialogue datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9241–9250, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.743. URL https://www.aclweb.org/anthology/2020.emnlp-main.743.

Y. Zhang, S. Sun, M. Galley, Y.-C. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, and B. Dolan. Dialogpt: Large-scale generative pre-training for conversational response generation. In *ACL, system demonstration*, 2020.

Z. Zhang, S. Ren, S. Liu, J. Wang, P. Chen, M. Li, M. Zhou, and E. Chen. Style transfer as unsupervised machine translation. *CoRR*, abs/1808.07894, 2018. URL http://arxiv.org/abs/1808.07894.

Y. Zheng, G. Chen, M. Huang, S. Liu, and X. Zhu. Personalized dialogue generation with diversified traits. *arXiv preprint arXiv:1901.09672*, 2019.

Appendix A

Supplementary material

A.1 Where to put these?

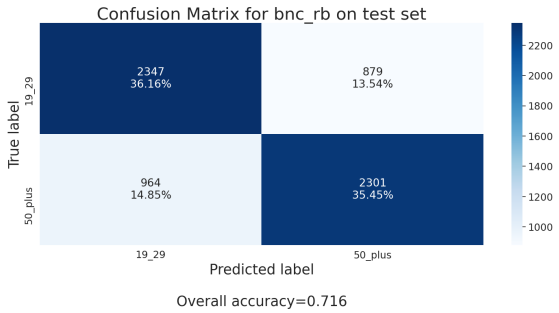


Figure A.1: Confusion matrix BERT age classifier on balanced BNC **test** set.

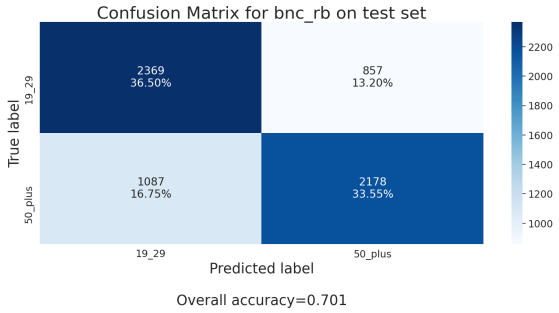


Figure A.2: Confusion matrix LSTM age classifier on balanced BNC **test** set.

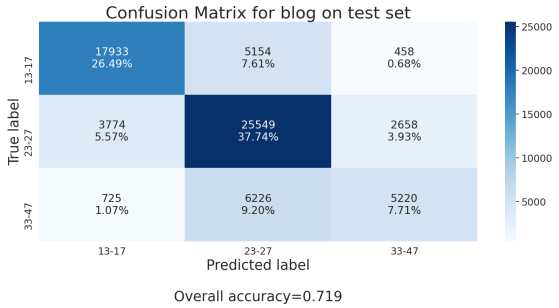


Figure A.3: Confusion matrix bi-LSTM age classifier on blog corpus **test** set.

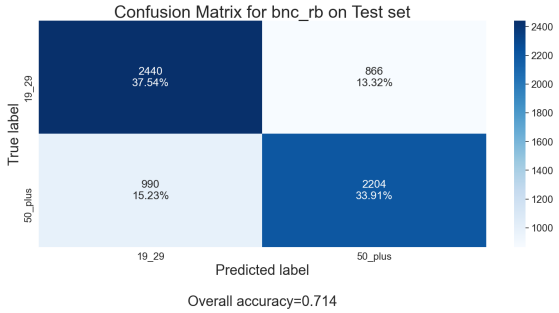


Figure A.4: Confusion matrix for best trigram age classifier on **balanced** BNC **test** set.

A.2 Age discrimination on the imbalanced British National Corpus

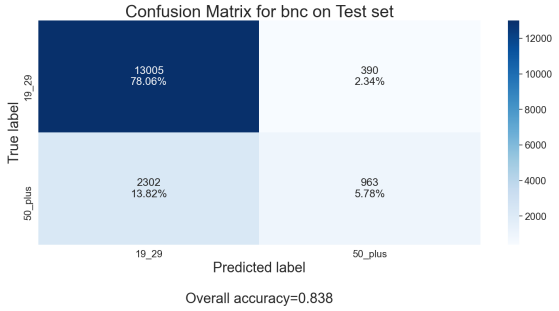


Figure A.5: Confusion matrix for best bigram age classifier on BNC test set.

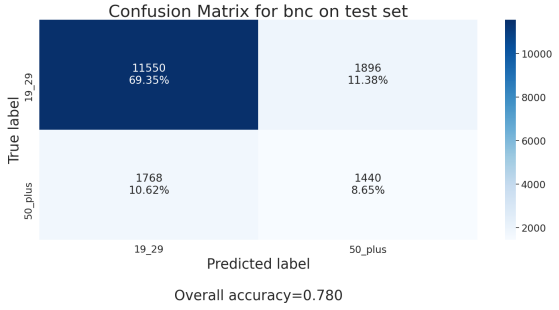


Figure A.6: Confusion matrix bi-LSTM age classifier on BNC test set.

	19-29		50+
coef.	**n-gram**	**coef.**	**n-gram**
-3.20	um	2.37	yes
-2.84	cool	2.12	you know
-2.58	s**t	2.09	wonderful
-2.12	hmm	1.90	how weird
-2.09	like	1.84	chinese
-2.02	was like	1.73	right
-1.96	love	1.71	building
-1.96	as well	1.66	right right
-1.88	as in	1.55	so erm
-1.84	cute	1.43	mm mm
-1.82	uni	1.41	cheers
-1.79	massive	1.39	shed
-1.79	wanna	1.37	pain
-1.79	f**k	1.36	we know
-1.72	tut	1.08	yeah exactly

Table A.1: [L: Including stopwords.] For each age group, top 15 most informative n-grams used by the trigram model. **coef.** is the coefficient (and sign) of the corresponding n-gram for the logistic regression model: the higher its absolute value, the higher the utterance's odds to belong to one age group. * indicates masking of foul language.

Model	**Accuracy** ↑ better	$\boldsymbol{F}_1^{(19-29)}$ ↑ better	$\boldsymbol{F}_1^{(50+)}$ ↑ better
Random	0.500	0.500	0.500
unigram	0.701 (0.007)	0.708 (0.009)	0.693 (0.004)
bigram	0.719 (0.002)	0.724 (0.003)	0.714 (0.003)
trigram	0.722 (0.001)	0.727 (0.003)	0.717 (0.001)
LSTM	0.693 (0.003)	0.696 (0.005)	0.691 (0.007)
BiLSTM	0.691 (0.009)	0.702 (0.017)	0.679 (0.007)
BERT$_{frozen}$	0.675 (0.003)	0.677 (0.008)	0.673 (0.010)
BERT$_{FT}$	**0.729** (0.002)	**0.730** (0.011)	**0.727** (0.010)

Table A.2: Dialogue dataset [L: Including stopwords.]. Test set results averaged over 5 random initializations. Format: *average metric (standard error)*. Values in **bold** are the highest in the column; in blue, the second highest.

Model	**ppl.** ↓ better	**Dist-1** ↑ better	**Dist-2** ↑ better	**Dist-3** ↑ better	**Acc.** ↑ better
Baseline***	27.45 (±7.27)	0.90 (±0.10)	0.92 (±0.05)	0.86 (±0.09)	57.5%
B_{100MCW}***	26.68 (±8.77)	0.89 (±0.10)	0.92 (±0.05)	0.86 (±0.09)	51.7%
$B_{Y,FB}$	27.11 (±7.45)	**0.91** (±0.09)	**0.92** (±0.04)	**0.87** (±0.09)	68.3%
$B_{O,FB}$	25.99 (±6.41)	0.88 (±0.11)	0.92 (±0.05)	0.86 (±0.09)	62.5%
$B_{Y,100MIU}$	28.48 (±11.96)	0.88 (±0.12)	0.91 (±0.06)	0.86 (±0.10)	69.2%
$B_{O,100MIU}$	**25.57** (±7.44)	0.88 (±0.11)	0.92 (±0.05)	**0.87** (±0.09)	58.3%
$D_{Y,GPT2}$	33.02 (±12.24)	0.85 (±0.16)	0.89 (±0.07)	0.83 (±0.12)	73.9%
$D_{O,GPT2}$	32.86 (±18.08)	0.80 (±0.21)	0.84 (±0.13)	0.79 (±0.19)	63.3%

Table A.3: [L: Excluding stopwords.] Results of age-controlled language generation. Perplexity is perplexity w.r.t. GPT-1. Dist-n is number of distinct n-grams normalized by text length, as a measure of diversity. Young and old accuracy are the assigned probabilities of belonging to the young or old age categories.