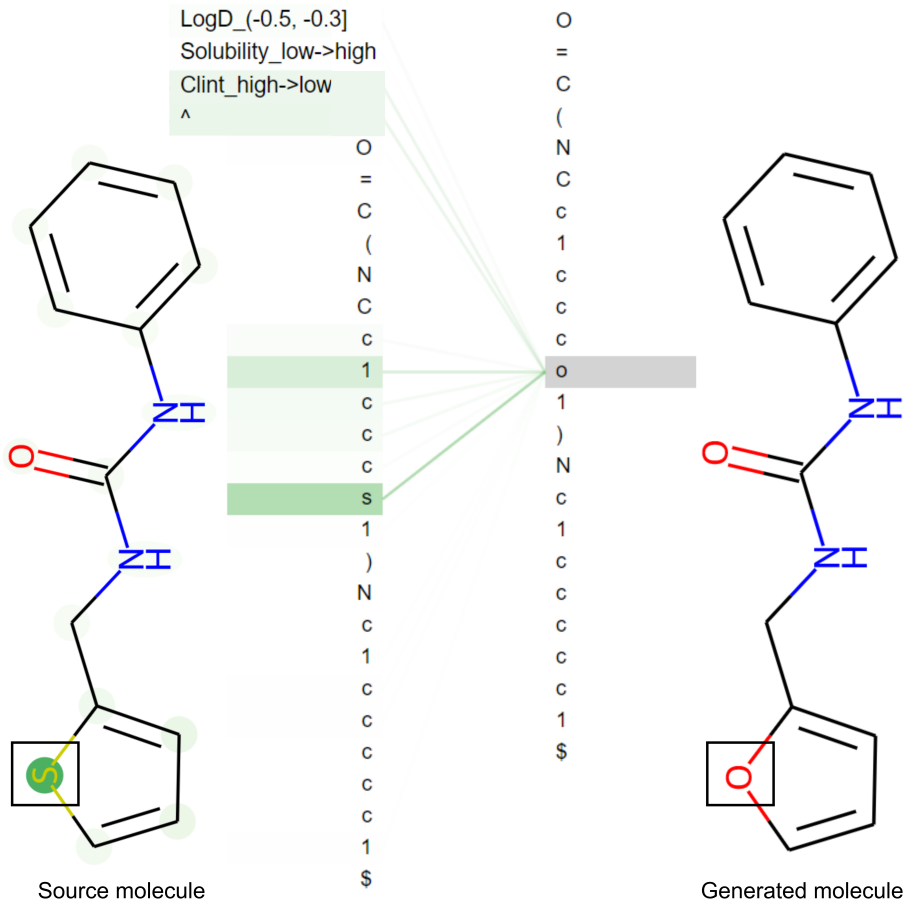


Explainable AI for the Transformer Model Used on Chemical Language

An Analysis of Attention in the Transformer when Applied on Molecular Optimization

Caroline Bükk
Linda Hoang

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2022

Master's thesis 2022

Explainable AI for the Transformer Model Used on Chemical Language

An Analysis of Attention in the Transformer when Applied on Molecular Optimization

Caroline Bükk
Linda Hoang

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2022

Explainable AI for the Transformer Model Used on Chemical Language
An Analysis of Attention in the Transformer when Applied on Molecular Optimization
Caroline Bükk
Linda Hoang

Supervisor: Richard Johansson, Department of Computer Science end Engineering
Supervisor: Jiazhen He, AstraZeneca
Examiner: Claes Strannegård, Department of Computer Science end Engineering

Master's Thesis 2022
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Cross-attention between a source and a generated molecule, from molecular optimization with the Transformer. Source molecule's SMILES: 'O=C(NCc1cccs1)Nc1ccccc1'. Generated molecule's SMILES: 'O=C(NCc1ccco1)Nc1ccccc1'.

Typeset in LaTeX
Gothenburg, Sweden 2022

Explainable AI for the Transformer Model Used on Chemical Language
An Analysis of Attention in the Transformer when Applied on Molecular Optimization
Caroline Bükk, Linda Hoang
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

One of the main challenges in drug discovery is to find new molecules with desirable properties. In recent years, using deep learning models to change the properties of a molecule has shown promising results. This task is done by letting the model transform the original molecule, and is often referred to as *molecular optimization*. A problem with using deep learning models is that it is difficult to understand what the model bases its decisions on. In our project, understanding what the model basis its decision on could be valuable feedback to drug designers and chemists. It could both extend their understanding of suitable transformations in different scenarios and provide insight in how the model could be improved.

In this thesis, we have focused on explaining the Transformer model, when used to perform molecular optimization. As the molecules in this task are expressed in a chemical language, this problem can be viewed as a machine translation problem. The predicted molecule then corresponds to the translation of the input molecule and the desirable property changes. To explain the model, we considered a set of assumptions of what the model would focus on. The assumptions were inspired by the chemists' intuition regarding what should influence the transformation most. The attention weights of the cross-attention layer were then analysed to test if these assumptions were correct. In order to determine if a contribution to the transformation could be considered important, relative comparisons between different parts of the input and output were used.

We found that in some regards, the chemists' intuition agreed with our comparisons of the attention weights. However, in some cases, the absolute value of the attention weights on the important parts were still very low. For future work, we suggest additional assumptions based on the chemists' intuition and experiments to test them. We also suggest to use the explainability technique, integrated gradient, that could be applied similarly and used to verify our results.

Keywords: Explainable AI, attention weights, transformer, NLP, molecular optimization, machine translation, machine learning

Acknowledgements

First of all, we would like to thank our supervisor at AstraZeneca, Jiazhen He, who has created the work we base our thesis on, and who has helped us with the project and always given us valuable knowledge and feedback. Also, we would like to thank our supervisor Richard Johansson from the Department of Computer Science and Engineering at Chalmers and GU for his guidance, good advice and feedback.

In addition, we would like to thank Ola Engqvist who has provided us with knowledge from a chemist's point of view and our opponents Clara Nyman and Christian Josefson for proofreading and giving feedback on the report. Last but not least, we want to thank our friends and family who have supported us during these sometimes stressful periods.

Caroline Bükk and Linda Hoang, Gothenburg, June 2022

Contents

Abbreviations

This is a list of abbreviation which are frequently used in the report.

ADMET - **A**bsorption, **D**istribution, **M**etabolism, **E**xcretion and **T**oxicity

MMP - **M**atched **M**olecular **P**airs

NLP - **N**atural **L**anguage **P**rocessing

SMILES - **S**implified **M**olecular-**I**nput **L**ine-**E**ntry **S**ystem

Notations

This is a list of notation which are frequently used in the report.

Core - The part of the molecule that remains constant during the transformation.

Cross-attention - The attention between the encoder and decoder.

R-group - The part of the molecule that is added, removed or transformed during the transformation.

List of Figures

1

Introduction

In this chapter, we will first provide a background for the process of drug discovery, molecular optimization and explainability methods, which all are central subjects for our thesis. We will then describe our aim with the thesis and present our research questions. Finally, we discuss the thesis limitations and provide an outline for the rest of the report.

1.1 Drug discovery

In drug discovery, finding the right compounds to treat diseases is a complex and lengthy process. Figure 1.1 shows a general overview of the drug discovery process. Most new drugs today arise from discovery programs that begin with identifying a biomolecular target that has a potential therapeutic value and then searching for drug-like compounds which typically selectively bind to the molecular target and interfere either with its activity as a receptor or enzyme. Molecular libraries are screened and resulting lead compounds are optimized again in a cycle of feature designs, synthesis, assaying of numerous analogues, and animal studies [6]. After that, the human clinical period starts. This is the period where the complex differences between animals and humans are addressed. Finally, the differences between humans are considered. This includes the field of pharmacognosy, which studies how a person's genes influence how they react to the drugs [1].

A lead compound is a chemical compound that has shown potential as a disease therapy and could lead to the development of a new medicine [7]. When the lead compound has been identified, the chemical structure is used as a starting point to make a drug that has as many benefits and as few harms as possible. The lead compound is optimized to improve potency and ADMET (Absorption, Distribution, Metabolism, Excretion, and Toxicity) [8] properties, which describes the disposition of a pharmaceutical compound within an organism, in other words "drug-like" properties [9]. The process requires iterative screening runs. With the expectation that

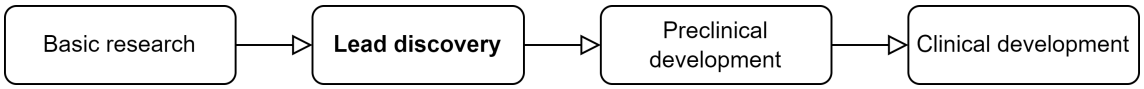


Figure 1.1: A general overview of the drug discovery process. The Figure is inspired by [1].

the properties of the new compounds will be improved, and favourable compounds will go forward to in *vitro* (in test tube) and in *vivo* (in living organism) testing. An example of some ADMET properties which are relevant for our thesis are logD, solubility and clearance. They will be explained in more detail in Section 2.4.1.

1.2 Molecular optimization

In the lead discovery process, starting with a promising molecule and transforming it to achieve a balance between different properties is a part of the optimization steps. This is typically done using the chemists' intuition for which transformations might give the desired properties. In recent years, other approaches have been explored for improving this process, such as using machine learning models [10, 11]. He et al. [3] used the Transformer model to transform molecules to make them satisfy a set of desirable properties. This task of transforming molecules using a deep learning model is called *molecular optimization* [3]. In the article, He et al. [3] represented the molecules in a string-based format called *Simplified Molecular Input Line Entry System* (SMILES) [12], which is described in depth in Section 2.4.2. By using this string-based format, the molecular optimization can be viewed as a classical machine translation problem.

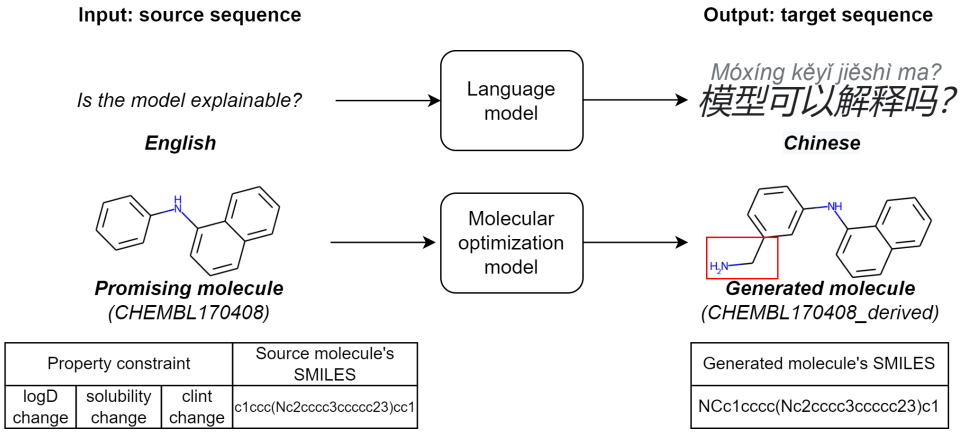


Figure 1.2: A comparison between machine translation and molecular optimization, where a promising molecule "translates" to a predicted molecule. Specifically, the input to the molecular optimization model are property changes: logD, solubility and clearance, concatenated with the source molecule's SMILES. The red marked box in the generated molecule shows the part added to the molecule.

To train the model, source and target data are used. The source data consists of the molecule before the transformation and a set of properties that the molecule should fulfil after the transformation. The target is the transformed molecule which fulfils the desirable properties. After training, the trained model is used for generating new molecules with the desired properties. In Figure 1.2 the similarities between the classical machine translation task and molecular optimization is visualized. For the translation task, the language model is given an English sentence and outputs

the translation in Chinese. For the molecular optimization task, the model is given a promising molecule and three property constraints. The model then outputs a transformed molecule which fulfils the property constraints.

The molecule can be divided into two parts: the core and the R-group. The core is the part that remains constant throughout the transformation and the R-group is the part that is added, removed or transformed in the transformation. As we are only interested in the cases where a transformation occur, the R-group can exist in three different ways: only in the source molecule, only in the predicted molecule and in both the source and predicted molecule. Figure 1.2 is an example of a prediction where the R-group only exists in the predicted molecule.

1.3 Explainability methods

Explainability methods is an umbrella term used to describe techniques which aim to explain the underlying decision mechanisms in deep learning models. They are often used to give further confidence in the model's predictions. Explaining refers to the ability of humans to understand the results of a solution generated by Artificial Intelligence [13]. One example of these so-called explainability techniques is *feature importance*. The main idea is to numerically describe how much different parts of the input contributed to generating different parts of the output.

In the machine learning field, there have been techniques developed for explaining natural language processing (NLP) [14]. NLP is the computerized approach to analyse text, spoken or written by humans, for a range of tasks or applications [15]. Attention scores and first-derivative saliency are two widely used methods for feature importance-based explanations within the field of NLP [14]. Text-based features are more intuitive for humans to interpret, which may explain the widespread use of attention-based approaches in the NLP domain. Nonetheless, these methods have not been deeply explored for chemical languages. In this project, the aim is to investigate attention scores as an explainability technique for the Transformer model when applied to chemical language.

In the field of molecular optimization, explainability techniques could be used to investigate what parts of the input that contribute to the transformation. Primarily, it would be valuable feedback for drug designers and medicinal chemists to know how the model designs the molecules the way it does and if it replicates the chemists' intuition regarding transformation of molecules.

1.4 Aim

The high-level-goal of the thesis is to investigate the application of explainability techniques, in particular attention weights in a Transformer model used for chemical language i.e., molecular optimization. Inspired by the chemists' intuition, we set up research questions regarding what part of the input that should affect the transformation in the generated molecule. By chemists' intuition, we refer to chemists with 5-10 years of work experience in field of drug design. The research questions can be divided into three main categories: property change, molecular structure and the transformed part in the source molecule. For each category, we investigate the effect on the transformation.

- **Property change:** The chemists' intuition regarding the property change is that it will be important for the transformation. As it is difficult to judge how important something is by just looking at an isolated number, we will use two comparisons. First, we will compare how much the other token categories affect the transformation. Then, we will compare how much the property tokens affect the transformation vs. how much it affects the reconstruction of the core of the molecule.

- **Molecular structure:** The chemists' intuition for the molecular structure is that the input atoms' contribution to generating the atoms in the generated molecule varies according to their distance to it. The distance that we will consider is the topological distance in the molecule. To define the topological distance, we consider the molecule to be a graph. Each atom is a node and each bond is an edge. The topological distance between two atoms is then the graph distance between the atoms.

  It is also believed that the importance of the atoms close to the transformation will contribute more to the transformation when the transformation is small. The reasoning behind this is that for small property changes, it will be easier for the model to accomplish the change by only focusing on a small part. However, for larger property changes, the model might need to focus more on the entire structure. Due to time constraints, this was omitted from the research questions and was not investigated in this thesis. However, a brief discussion of how this could be done can be found in the future work section in the conclusion, Chapter 5.

- **Transformed part in source molecule:** In the molecular pairs, where both the source and generated molecule have an R-group, it might be reasonable to think that the R-group in the source molecule will contribute more than the average atom, to the transformation. The reasoning for this is that omitting or changing a part in the source molecule will likely also change the properties of the molecule, and will thereby need to coordinate with the generation of the R-group. To determine the relative importance of the R-group in the source molecule, we will compare its contribution to the R-group in the predicted molecule with the contribution of the core atoms in the source molecule.

1.4.1 Research questions

All of these hypotheses are specified in the following research questions.

1. Is the property change important for generating the transformation?
2. Is the transformation affected most by the atoms closest to its corresponding part in the source molecule?
3. Is the transformed part in the source molecule important for generating the transformation?

1.5 Limitations

For the Transformer model, the most commonly used explainability technique is to look at the attention weights between the encoder and decoder, called cross-attention [16]. However, the attention scores of other parts of the Transformer and gradient based methods are also widely used explainability techniques. Due to time constraints, we will only use attention and in particular the *cross-attention* weights to answer our research questions. The cross-attention will be explained in Section 2.3.

Regarding the chemists' intuition, only the intuition covered in the research questions will be investigated. However, approaches to investigate more assumptions based on the chemists' intuition will be described under future work in the conclusion chapter.

1.6 Thesis outline

In this first chapter, we have given a brief introduction to the field of drug discovery and some well known explainability methods. We have also defined our research space and the limitations of the thesis. We will now provide a brief overview of the rest of the thesis.

- In Chapter 2, we will give an introduction to optimization in drug discovery, an in-depth background to molecular optimization, sequence-to-sequence models, the Transformer model and attention as an explainability method.
- In Chapter 3, we will present our methodology for the data preparation, describe the implementation of the model and explain how the data retrieval for the explainability analysis was performed.
- In Chapter 4, we will present our results and discuss its implications for the research questions.
- In Chapter 5, the main conclusions are presented and some approaches for future work are suggested.

2

Theory

In this chapter, we have provided a background for all the topics this thesis will address. We begin with describing *lead optimization*. Then, we introduce sequence-to-sequence models with recurrent neural networks and today's most common architecture for sequence-to-sequence problems, *the Transformer model*. We first describe the architecture of the Transformer model in detail and then explain how the Transformer model is used to perform molecular optimization. In the last section we present explainable AI for NLP, describe why models need to be explained, and how attention can be used as an explainability technique.

2.1 Lead optimization

In the early stage of drug discovery, identifying compounds that show promise as a treatment for a disease and can lead to the development of a new drug is the main goal. These compounds are so-called *lead compounds* and are later tested in further clinical phases. Figure 2.1 shows an overview of the drug discovery process. One begins with the search of compounds that bind to a molecular target as a receptor or enzyme (target identification). Following that, screening processes such as high throughput screening (HTS) is done, where the entire compound library is screened directly against the drug target and generates hit compounds [1]. Then, in the hit-to-lead phase, the hit compounds are evaluated and undergo limited optimization to identify promising lead compounds. The lead compounds are then also optimized. This optimization process involves multiple rounds of synthesis and characterization of potential drugs to develop a picture of how chemical structure and activity relate to their target and metabolism interactions. E.g., leads are opti-

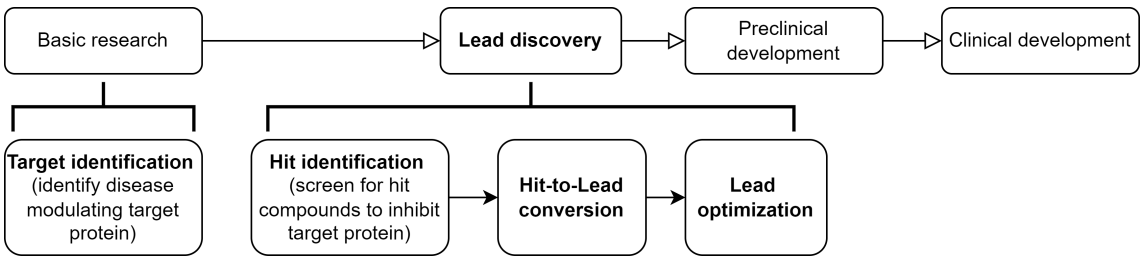


Figure 2.1: A more detailed overview of the drug discovery process. The Figure is inspired by [1, 2].

mized to fulfil desirable properties such as physicochemical properties and ADMET (Absorption, Distribution, Metabolism, Excretion and Toxicity) properties. After the lead optimization, the preclinical phase begins [1].

When trying to find a new drug, it is highly unlikely to randomly find a suitable molecule as the drug-like molecule space exceeds 10^{23} molecules [17]. A common approach is therefore to start with a promising molecule (like the lead compound) that lacks some desirable properties. This molecule is then optimized through a transformation, often using the chemists' intuition for which transformation that might achieve the desirable properties.

Recent development in machine learning has enabled deep learning models to be trained for the specific task of transformation, without using the chemists' intuition [10, 11]. In a recent article by He et al. [3], a deep learning model which can be trained to transform molecules so that they satisfy some desirable properties was presented. This task is referred to as *molecular optimization*. The molecular optimization problem is similar to a machine translation problem in NLP. A promising molecule, represented as a string, is translated into a similar molecule with optimized properties, much like how a sentence in English would be translated into a sentence in Chinese see Figure 1.2. More specifically, the work by He et al. [3] showed potential for a model based on the Transformer to perform molecular optimization. The Transformer is a neural network architecture used for sequence-to-sequence tasks in NLP.

2.2 Sequence-to-sequence models

Sequence-to-sequence (seq2seq) learning takes in an input sequence of words and generates an output sequence of words. There are different applications of this such as language translation, text summarization, conversational bots and image captioning. This also means that the length of the input sequence is not necessary equal to the length of the output sequence, e.g., the input-to-output can be one-to-many or many-to-one. Seq2seq models commonly uses recurrent neural network (RNN), which are a variant of feedforward neural networks that can handle sequential data and be trained to remember information from the past. At each time step t, the RNN uses a so-called *hidden states*, h_t. The network stores the state of information of the previous input to help generate the next output of the sequence. It does this by the recurrence formula (i.e., feedback loop):

$$h_t = f_W(h_{t-1}, x_t), \tag{2.1}$$

where h_{t-1} is the previous hidden state, x_t is the input vector at some time step, f_W is some function with parameter W, and h_t is the new hidden state [18]. In particular, the classical ("vanilla") RNN model uses an activation function (e.g., tanh) for the hidden vector h_t and then output y_t

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t) \tag{2.2}$$

$$y_t = W_{hy}h_t. \tag{2.3}$$

Deep RNNs have multiple layers at each time-step and re-uses the same weight matrix at every time-step. One problem RNN faces is short-term memory loss, meaning the model is not able to memorize data for a long time and starts to forget its previous input [18]. This is caused due to the vanishing gradient problem. During backpropagation through time, at each step, the gradient is calculated. If the gradient of the previous layer is small, then the gradient of the current layer will be even smaller. Too small gradients will not update the weights, and therefore the network will not learn. The vanishing gradient issue has been solved with other types of RNNs: LSTM (Long Short-Term Memory) [19] and GRU (Gated Recurrent Units) [20] networks, where LSTM is a generalization of GRU. These two networks have a more complex hidden unit computation. They use memory cells to store activation values (signals to activate the neuron) of previous words in long sequences. So-called "*gates*" are used for controlling flow of information in a network. The gates learn which input in the sequence that is important and store their information in the memory unit.

In 2014 Cho et al. [21], presented the encoder-decoder architecture for seq2seqs problems, where one RNN encodes a sequence of symbols into a fixed-length vector representation (i.e., context vector), and the other decodes the representation into another sequence of symbols. The best model at the time was to combine the encoder-decoder architecture with the attention mechanism [22]. The attention mechanism is a part of a neural architecture that enables the model to dynamically highlight relevant features of the input data. The core idea is to compute a weight distribution on the input sequence and assigning higher values to more relevant elements [23]. The currently most used architecture for seq2seq tasks is the Transformer [16].

2.3 The Transformer model

The Transformer uses the encoder-decoder architecture and is based on the attention mechanism [22], which emphasizes certain elements more than others. There are three ways in which attention occurs in the model: *self-attention* in the encoder, *self-attention* in the decoder and *cross-attention* between the encoder and decoder [16]. See the red attention blocks in Figure 2.2.

Self-attention relates different positions of a single sequence in order to compute a representation of the same sequence. As an example, let the following sentence be the input sentence, which we want to translate: "*The animal did not eat the food because it was too full*". Does "*it*" refers to the food or to the animal? For a human it might be easy to understand, but not for an algorithm. When the model is processing the word "*it*", self-attention allows it to associate "*it*" with "*animal*". As the model processes each word (each position in the input sequence), self-attention allows it to look at other positions in the input sequence for information that can help lead to a better encoding for this word.

Cross-attention relates two different sequences, and gives information from the input sequence to the decoding layers, such that the decoder can predict the next sequence

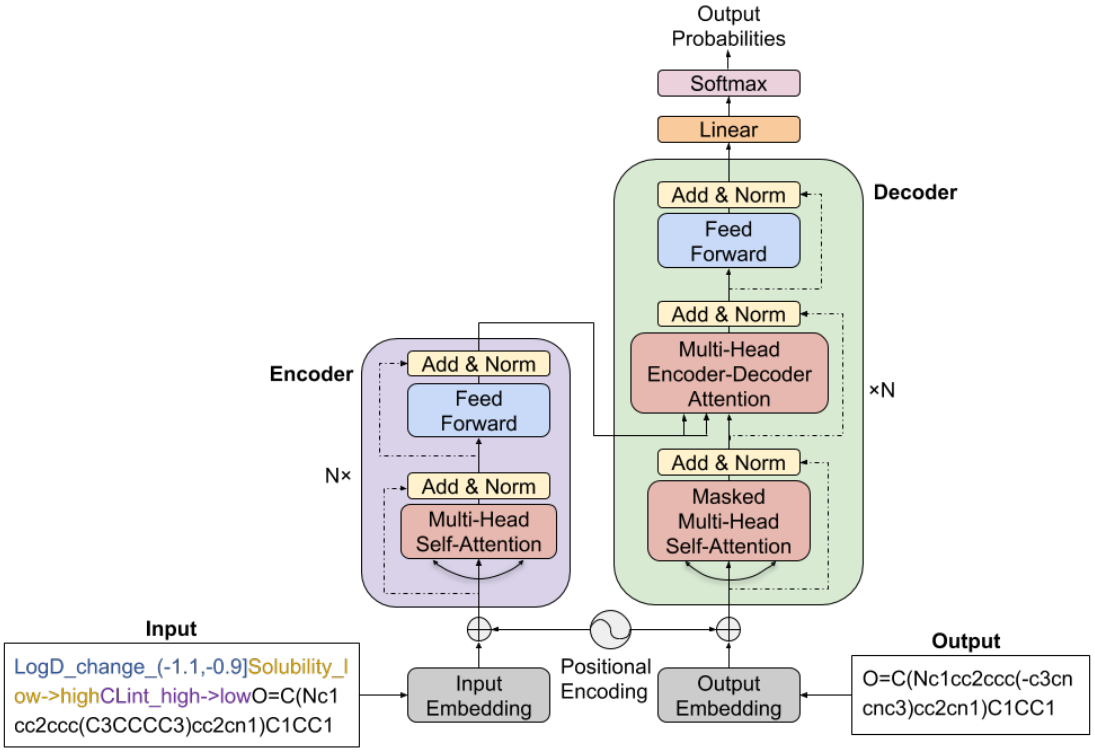


Figure 2.2: A visualization of the Transformer model architecture. The input and output sequences are specifically for the molecular optimization task. Figure source [3].

token. The next token is then added to the output sequence. In this project we will only look at the cross-attention since it has a more direct relation to the output sequence.

2.3.1 Input embedding

The first step in the encoder is to convert each word in the sentence to a word embedding. An embedding layer is essentially a lookup table to find a learned vector representation of each word. The next step is to combine the positional encoding information with the embeddings.

2.3.1.1 Word embedding

Firstly, one represents each word of the input sentence as a one-hot encoding vector. This is a vector with all elements set to zero, except for the element representing the encoding word, which is one. The length of the vector is determined by the size of the vocabulary. These one-hot encoding vectors are very sparse. By instead multiplying the one-hot vectors with a learned weight matrix W, one obtains a real-valued vector, a so-called *word embedding*. In the original Transformer article, an embedding dimension of size 512 is chosen. The weight matrix has the shape [vocabulary size, embedding dimension]. The Transformer uses a random initialization of the weight matrix and updates these weights during training, i.e., learning its

own word embeddings.

2.3.1.2 Positional embedding

Since the model does not use recurrence or convolution to learn the sequential information, the model must get some information about the relative or absolute position of tokens in the sequence. Therefore, *positional encodings* are added at the bottom of the encoder and decoder stacks. There are many choices of positional encodings. For example, the sine and cosine function of different frequencies can be used:

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}}) \tag{2.4}$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}}), \tag{2.5}$$

where *pos* is the position and i is the dimension [16]. These functions have linear properties, which the model easily learns to attend to.

2.3.2 Scaled dot product attention

An attention head takes a sequence of vectors $x = [x_1, ..., x_n]$ as input, where n is the number of input tokens. Each vector x_i is transformed into query, key, and value vectors q_i , k_i , v_i with separate linear transformations. Intuitively, one can think of the query as a representation of what kind of information that one is looking for. The key represent the relevance to the query, and the value represent the actual contents of the input. The head computes the attention weights α between all word-pairs as a softmax-normalized dot product between the query and key vectors, normalizing the weights to a value between 0 and 1, where d_k is the dimension of keys. The output o of the attention head is a weighted sum of the value vectors.

$$\alpha_{ij} = \frac{\exp(q_i^T k_j/\sqrt{d_k})}{\sum_{l=1}^{n} \exp(q_i^T k_l/\sqrt{d_k}} \tag{2.6}$$

$$o_i = \sum_{j=1}^{n} \alpha_{ij} v_j \tag{2.7}$$

In practice, the attention function is computed on a set of queries, keys, and values simultaneously, packed together into a matrix Q, K, and V.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V, \tag{2.8}$$

where the dot product of the queries with all keys are normalized with $\sqrt{d_k}$ [16].

In self-attention, keys, values, and queries are generated from the same sequence. In cross-attention, the queries are generated by a different sequence than the key-value pairs. The attention function corresponds to computing one head. To perform the attention function in parallel, the model uses multiple heads. [16]

2.3.3 Multi-head attention

Instead of performing single attention functions, the multi-head attention computes attention to capture various aspects of the input. This allows the model to obtain information from different sub-spaces. An example of what different heads can focus on is shown in Figure 2.3.

When learning the various representations, each head is a unique linear projection of the input representation as query, key, and value. The scaled dot product attention is calculated h times in parallel and the outputs are concatenated. One linear projection is applied by:

$$\begin{aligned} MultiHead(Q, K, V) &= Concat(head_1, ..., head_h)W^O \\ head_i &= Attention(QW_i^Q, KW_i^K, VW_i^V). \end{aligned} \tag{2.9}$$

The projections are parameter matrices W_i^Q, W_i^K and W_i^V and W_i^O. [16]

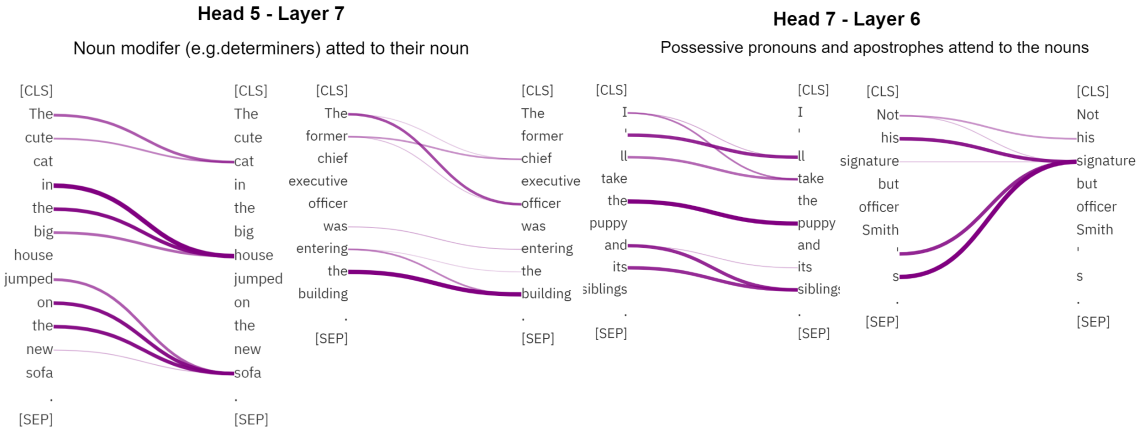


Figure 2.3: An example of how the transformer-based model BERT attention heads focus on different parts, which corresponds to different linguistic phenomena. The figures are inspired by [4].

2.3.4 Overall model architecture

Like sequence-to-sequence models, the Transformer uses an encoder-decoder architecture. Both encoder and decoder are stacked on top of each other, creating N identical layers. Figure 2.2 describes the Transformer architecture for one encoder-decoder layer. The encoder is composed of two sublayers. The first is a multi-head self-attention mechanism and the second is a position-wise fully connected feed-forward network. There is a residual connection around each two sublayers, followed by normalization. The output of each sublayer is $LayerNorm(x + Sublayer(x))$, where $Sublayer(x)$ is the function implemented by the sublayer itself. The dimension of all sublayers in the original transformer model and embedding layers is $d_{model} = 512$. The decoder, in addition to the two sublayers in each encoder layer, has a third sublayer, which performs multi-head attention over the output of the encoder stack. To prevent positions in the decoder from attending to future positions,

the self-attention layer masks the future positions. This ensures that the prediction for position i can only depend on the known outputs at positions less than i. [16]

2.4 Molecular optimization with the Transformer

Our thesis is based on the molecular optimization model from the article[1] [3]. In the article, the molecular optimization was performed with the original Transformer model[2] [16] described in Section 2.3. It was trained using a source molecule and a set of desirable properties as input and a molecule with a small transformation that fulfilled these properties as target. By only using closely related molecules, the aim was to learn the model to choose the same kinds of transformation that a chemist might suggest to improve the properties of the molecule.

How the model is used to perform molecular optimization will be explained in more detail in Section 2.4.5. Before that, we will introduce some concepts to make this explanation easier to understand.

2.4.1 Molecular properties

In the molecular optimization, three important drug properties were considered, namely *clearance* (cl_{int} or clint), *logD* and *solubility*.

Clearance - Clearance or more specifically intrinsic clearance is the ability of the liver to remove drugs independent of other physiological factors such as the liver blood flow or drug binding in the blood. The intrinsic clearance is the proportionality constant between rate of metabolism and the drug substrate concentration at the enzyme site. In other words, it estimates the compound's metabolic stability. [24]

LogD - LogD is a measure of lipophilicity of the molecule, in other words its potential to dissolve in lipids, fats and non-polar solvents. In drug discovery, it regulates the movement across membranes. A high lipophilicity is more likely to penetrate cell membrane, however, too high lipophilicity can be toxic. [25]

Solubility - The solubility is the ability for a drug to be dissolved in an aqueous medium and affects the absorption (the process of a drug moving from its site of delivery into the bloodstream) and bioavailability (the extent and rate at which the active drug enters systemic circulation). [26]

Clearance and solubility were quantified as either high or low and the desirable change was expressed as either *low* $\rightarrow$ *high*, *high* $\rightarrow$ *low* or *no change*. The logD is represented as its numeric value, and the change is described as an interval of how much it should increase or decrease.

These three property changes were tokenized to represent the first three tokens in the input sequence, see the first (coloured) part of the input to the Transformer

[1] Git repository: `https://github.com/MolecularAI/deep-molecular-optimization`

[2] The model used is identical to the original Transformer model, except that the input and output encoding dimensions were changed from 512 to 256, and label smoothing was changed from 0.1 to 0.

model architecture in Figure 2.2.

2.4.2 SMILES

For the model to understand the molecules, they are written in Simplified Molecular Input Line Entry System (SMILES) format [12]. SMILES is a chemical language that builds on chemical formulas. All SMILES only represent a single chemical structure, although a chemical structure can sometimes be written as many different SMILES. Note that the SMILES string does not only include atom characters, but also numbers representing start and end points of rings, brackets and other characters.

2.4.3 MMPs, core and R-group

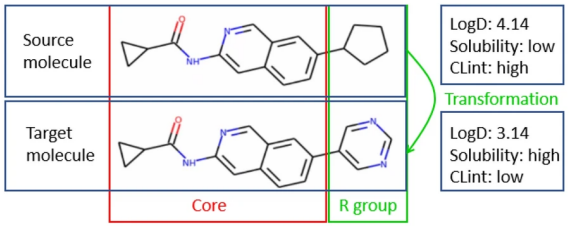


Figure 2.4: A matched molecular pair (MMP), containing source- and target molecule, where the structure is similar (core) but having one part being transformed, also known as the R-group. In this example, logD will *decrease* in value, solubility will change from *low->high,* and clearance will go from *high->low.* Figure source [3].

A *matched molecular pair* (MMP) is a pair of molecules that only differs by a minor single point change [27]. Figure 2.4 shows a matched molecule pair, containing a source and target molecule. The matched or constant part of the molecules is called the *core*, and the transforming part is called the *R-group*. One remark is that there is no requisite to have an R-group in both source and target molecules, i.e., the R-group in either source or target molecule does not have to exist.

2.4.4 Data preparation for molecular optimization

The properties: logD, solubility and clearance were predicted for each molecule using property prediction models, trained on AstraZeneca in-house experimental data [3].

The data used for training the model contained MMPs from the ChEMBL database [28], which is a manually curated database of bioactive molecules with drug-like properties. MMPs were extracted from the dataset, resulting in 9,927,876 MMPs that meet a set of constraints [3]. From these, 2 % were randomly sampled to be in the dataset. The dataset was first divided into training and validation set (90 %) and test set (10 %). The training and validation set was then further divided into training set (90 %) and validation set (10 %). This yielded a training set of 160,831 MMPs, a validation set of 17,871 MMPs and a test set of 19,856 MMPs.

2.4.5 Model summary

Using a chemical language to represent the molecules allows the molecular optimization problem to be viewed as a machine translation problem, where the source molecule together with the desired property changes is translated into the target molecule. The Transformer, which is the most commonly used model in machine translation, was used to predict the transformed molecules.

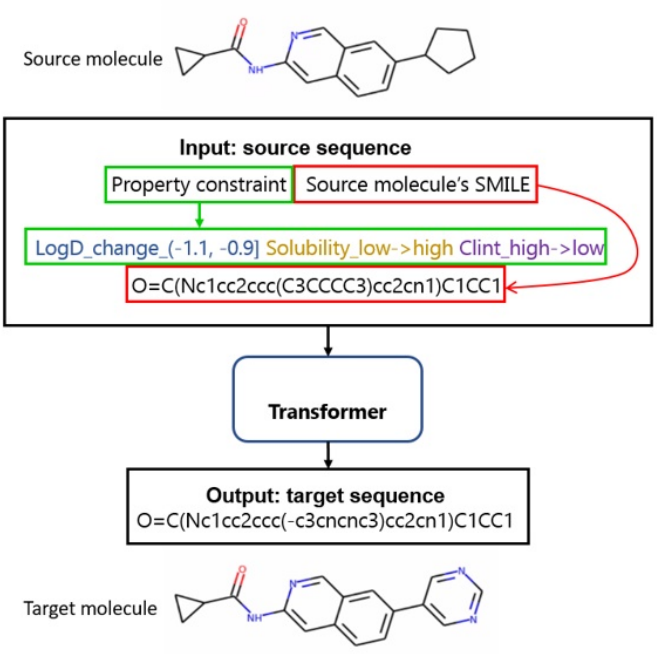


Figure 2.5: An example of how the Transformer model performs molecular optimization. During training, the source and target molecules come from an MMP, where the structures are similar to each other. The source molecule's SMILES string is concatenated with the property change between source and target molecules. The output from the model is the predicted molecule's SMILES string. Figure source [3].

The training was performed using the desirable property changes concatenated with the source molecule and used as input to the model. The output is a transformed molecule which should fulfil the property constraints. This flow is visualized in Figure 2.5.

The Transformer was trained for 60 epochs on 160,831 MMPs retrieved from ChEMBL and their properties were computed by a property prediction model. In the test set, about half of all predicted molecules satisfied all desired properties. Of this half, about 90 % of the source and target molecules only differ by a single transformation and no more than 1/3 of the molecules had changed.

2.5 Explainable AI for NLP

Deep learning algorithms have achieved high accuracy in complex domains such as natural language processing. These nested non-linear model structures can be compared with black-boxes, which allows you to see the input and output, but provide no information about what causes the models to arrive at their predictions [29]. Explainable AI is a set of tools and framework to help understand and interpret deep learning models. When a model is called explainable, it refers to the ability of humans to understand the results of a solution generated by Artificial Intelligence [13]. Explainability can focus both on which role different parts of the model play in learning the specific tasks and the importance the different input features play in the prediction.

2.5.1 Explainable AI in our project

In our specific case, the aim of the deep learning model is to ease the first steps of drug discovery. In the decision process of whether the model's molecule predictions are worth further investigation, it is important to understand why the model predicted them. If the model seems to value the same parts of the input in accordance with the chemists' intuition, it could justify continued research of the predicted molecules.

Moreover, the explainability of a well performing deep learning model could also be used to widen the intuition within the field. In our case, the chemical intuition for which transformations that are suitable in different scenarios could be altered by an explanation of how the model works.

To analyse the attention weights of the Transformer is a commonly used explainability technique within the field of NLP [30, 31]. Therefore, this seemed like a good starting point when applying explainability techniques on molecular optimization.

2.5.2 Attention as explanation

To understand which part of the input that is most significant to ensure that a prediction is made, it is common to attribute importance between different parts of the input and output. For the Transformer model, the most commonly used method for explanation is to look at the attention weights between the input and output (the cross-attention). In Section 2.3.2-2.3.3 it is shown how the attention weights for the Transformer were calculated. The cross-attention is the attention between the encoder and the decoder layer, where queries come from the previous decoder layer and the keys and values come from the encoder. The decoder then attends over all positions in the input sequence and shows which input tokens are most relevant for the predictions.

The attention mechanisms have seen widespread adoption in NLP models. In addition to improving predictive performance, it provides a distribution over attended-to input units, which shows the relative importance of different parts of the inputs. However, there is a debate in the NLP community whether attention can be used

as an explainability method. Bastings et al. [32] argues that input saliency methods, e.g., gradient-based methods, are better suited than attention when the goal is to find relevant input tokens to a prediction. Jain et al. [33] found that learned attention weights are often not correlated with gradient-based measures of feature importance, and that one can find different attention weights that results in the same prediction. Wiegreffe et al. [34] challenges the underlying assumptions in the work in [33], arguing that such a claim depends on one's definition of explanation. Further, Wiegreffe et al. [34] stated that further investigation is needed to take all elements of the model into account. They instead proposed alternative tests which resulted in meaningful interpretation of the attention mechanisms.

2.5.3 Visualization techniques

Another category within the explainable AI community is visualization techniques. The most widely used technique is saliency visualization techniques. Note that this meaning of saliency is for visualization purposes only, and should not get confused with saliency methods mentioned in Section 2.5.2, which is a competing explanation method to attention. There is not a general accepted agreement in the community how to use the word saliency. In our thesis, attention is not a saliency method, however attention heatmaps would be classified as a saliency visualization, based on the survey article of explainable AI for NLP by Danilevsky et al. [14].

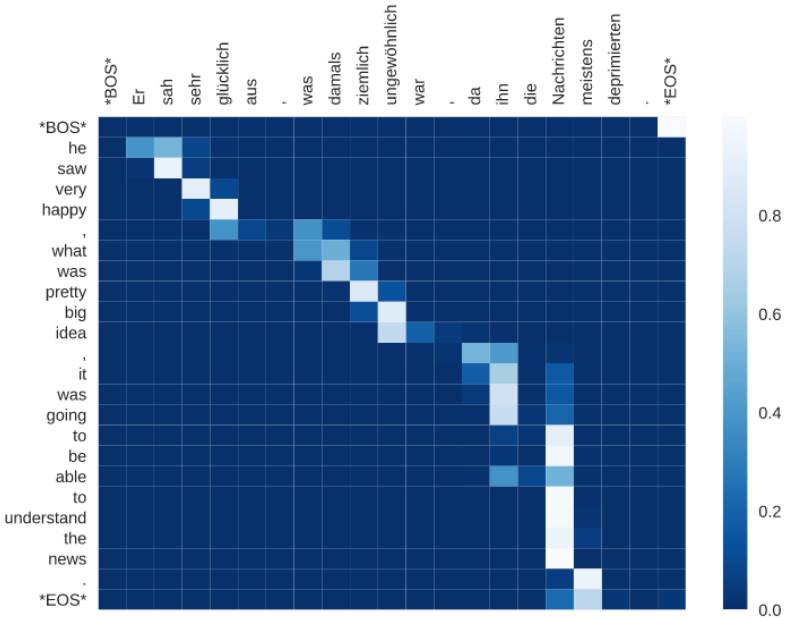


Figure 2.6: An attention heatmap of a translation from German to English. In this example, the generation of the translation to English has some errors. Figure source [5].

An example of saliency visualization is the saliency heatmap, also known as attention heatmap, which shows the input-output word alignment [22]. The heatmap shows which part of the input sequence that has the model's attention during the translation, see Figure 2.6. Danilevsky et al. [14] observed a strong correspondence

between feature importance-based explainability and saliency-based visualizations. All papers presented in the survey article [14] used feature importance to generate explanations and also chose saliency-based visualization techniques. The technique is popular because they present visually intuitive explanations and can easily be understood by different types of end users.

3

Methods

In our project, we have investigated attention as explainability technique on the molecular optimization model described in Section 2.4. In this chapter, we will describe everything we needed to do to obtain our results. We will begin by describing the data preparation, where we describe both how the data in the dataset were created and how the attention weights in the cross-attention were extracted. When all preparations have been described, we will provide a detailed explanation of all experimental setups, used to answer our research questions.

3.1 Data preparation

Here, we will describe the necessary preparations regarding data extraction. First, we will describe how the data in the dataset was chosen. We will then explain how this dataset was divided into subdatasets. Finally, we describe how the attention weights for each head in the cross-attention were extracted from the Transformer model.

3.1.1 Dataset creation

In this project, we were given a fully trained model and a dataset containing the source molecules and their corresponding desirable property changes. We used this dataset and the model to generate new transformed molecules, i.e., model predictions. For each source molecule in the dataset, ten transformed molecules were generated.

In the investigation of our research questions, we were only interested in using molecule pairs where the transformed molecules had the desirable properties and the transformation were not larger than a third of the molecule. The specific condition that the transformation should not be larger than a third of the molecule was chosen, as the model was only trained on such molecule pairs. To obtain a dataset with only the molecule pairs that we were interested in, we did a number of filtrations. First, we checked if the property constraint were fulfilled. The molecule pairs where the transformed molecule did not satisfy the property constraints were filtered out. Secondly, we checked that the molecule pair contained only a single and small transformation. We did this by comparing each molecular pair from the remaining dataset with a reference dataset with correct MMP transformations. The

reference dataset acted as a lookup table where you search for an MMP and retrieve the transformation. If the MMP was included in the reference dataset, we retrieved the transformation from the reference dataset. After this filtration, we obtained the dataset that will hence forth be referred to as the *full dataset*, and different subsets of it were used for the different experiments. (In Appendix A.1, Figure A.1 a sample of the *full dataset* is shown.)

3.1.2 Datasets for the experiments

For the experiments, two different subsets of the *full dataset* were used. In the first dataset, we randomly sampled 10 000 samples from the *full dataset*. This dataset will hence forth be referred to as the *general dataset*. In the second dataset, we filtered the *full dataset* to only contain samples that had an R-group in both the source and generated molecule. After this filtration, we randomly sampled 10 000 samples. This dataset will hence forth be called the *R-group dataset*.

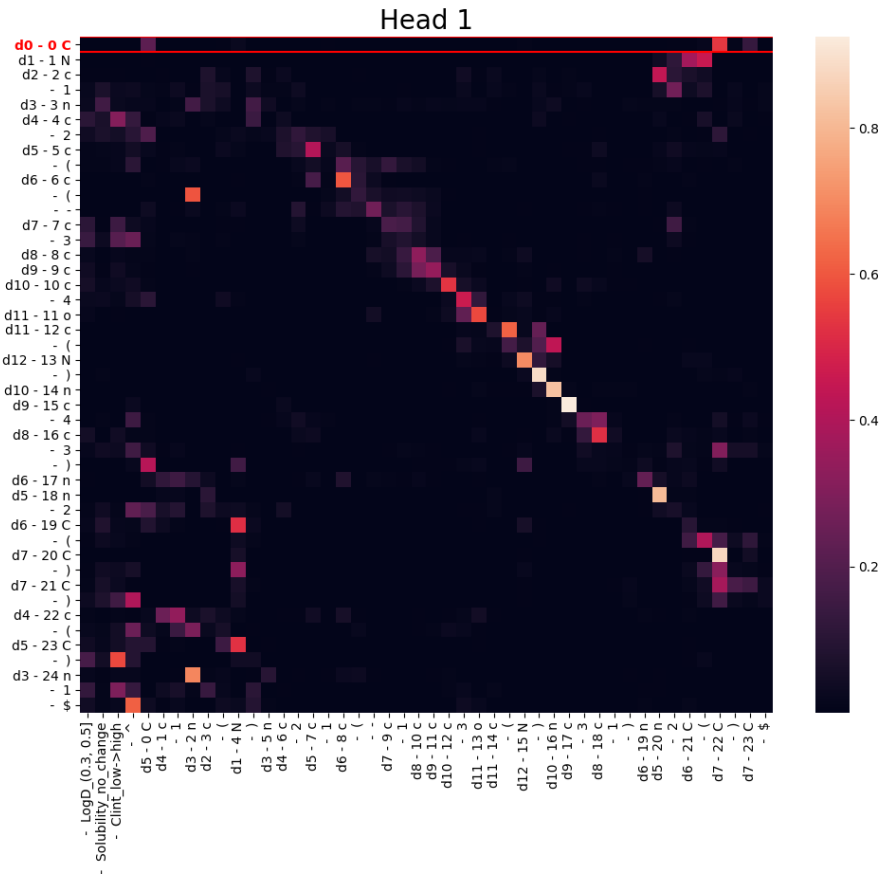
3.1.3 Extraction of attention weights

To obtain the cross-attention for each attention head, we first loaded the input (source molecule concatenated with desired properties) and output (generated molecule) into the model. Then, a forward pass for the data sample was made. The last layer of the cross-attention for each head was then obtained for each data sample.

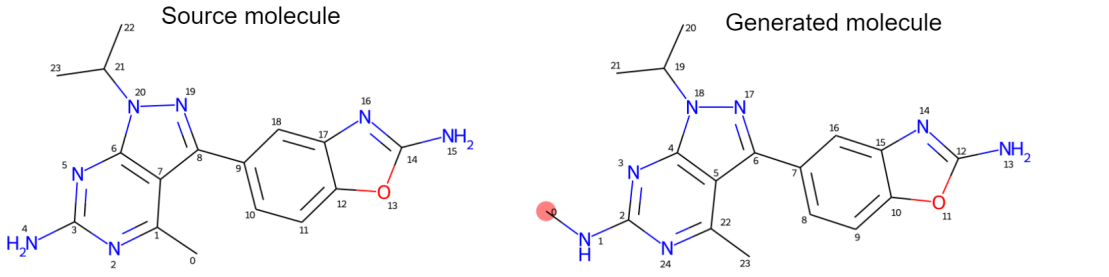
Each attention head has the dimension number of output tokens times the number of input tokens. This means that their size varies depending on the input and output. When examining the research questions, we considered each attention head individually.

Figure 3.1 shows an example of an attention heatmap, for a transformation of the source molecule's SMILES *'Cc1nc(N)nc2c1c(-c1ccc3oc(N)nc3c1)nn2C(C)C'* into the generated molecule's SMILES *'CNc1nc2c(c(-c3ccc4oc(N)nc4c3)nn2C(C)C)c(C)n1'*. For a visualization of all eight heads, see Figure B.1 and B.2 in Appendix B. The horizontal axis label shows each token of the three property changes concatenated with the source molecule's SMILES tokens. The vertical axis label shows each token of the generated molecule's SMILES tokens. The source and generated molecules are drawn in Figure 3.1b. The red highlighted bar represents the R-group in the generated molecule.

When we describe how we retrieved attention weights, we will often talk about the attention weights between a certain input and output token. With this, we simply mean the attention weights in the matrix that corresponds to the column of the input token and the row of the output token.



(a) An example of an attention heatmap with the input source molecule on the horizontal axis and the output generated molecule on the vertical axis. The red highlighted row is the R-group for the generated molecule. This example shows a single transformation between hydrogen H to carbon C, which is a common transformation in the dataset. In both axis labels there are two indexings. The index closest to the token is the regular index from 0 to n, n being the length of the molecule. The second index 'd_' stands for the topological distance to the R-group.



(b) The source- and generated molecule converted to SMILES strings. In the generated molecule, there is a red highlighted carbon atom, which is the atom in the R-group. The numbers in the molecules represent the atom indices of each respective molecule.

Figure 3.1: Attention heatmap (a) with input and output molecules, where the molecular structure are drawn in (b).

3.2 Experimental setup

In this section, we will provide a description for how all data used in the result section was retrieved. We will begin with a general overview of how we mapped atoms to their corresponding attention weights (Section 3.2.1). We will then move on to briefly describe how attention weights for different token categories were extracted (Section 3.2.2). After this, we will be ready to present a detailed description of how the attention weights for each research question category: property change, molecular structure and transformations groups are retrieved (Section 3.2.3-3.2.5).

3.2.1 Mapping of atom to attention weights

In Figure 3.2 the procedure for mapping atoms to attention weights are shown step-by-step.

- **Step 1: Match core atoms**
  The common core of atoms between the two molecules were found using an open-source cheminformatics software called RDKit[1] and an open-source MMP database tool[2]. When using the core as input to the function *GetSubstructMatches()* the core atom indices for both the source and generated molecules were returned in two different lists. The core part itself is the same in both the source and generated molecule, due to them being an MMP, however it can have different atom indices in the two lists. The two lists are aligned with each other, meaning that the same position in the two list corresponds to the same atom, e.g., the first element in both lists represent the same core atom.[3]

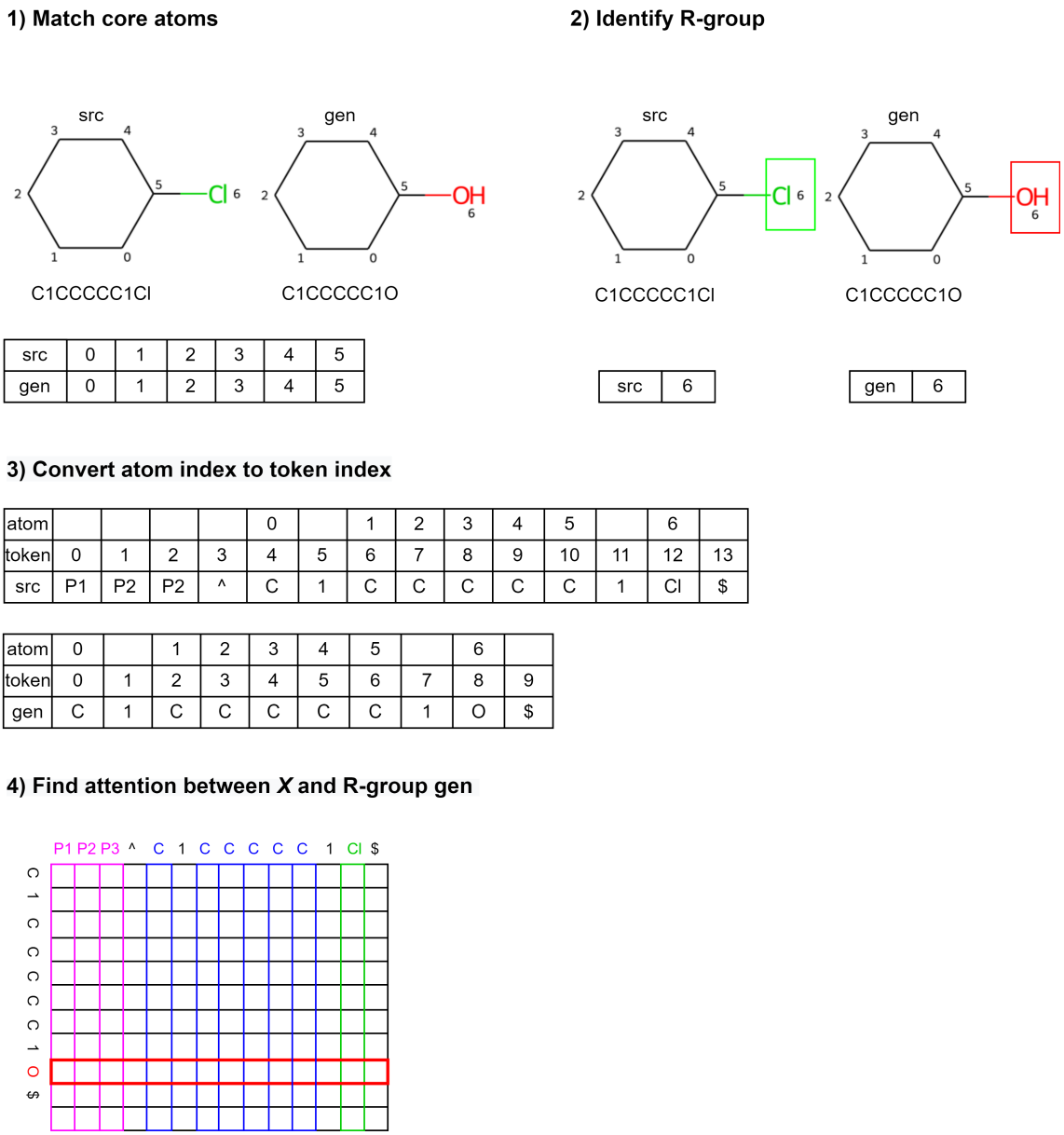
- **Step 2: Identify R-group**
  The R-group was found by first identifying all atoms in the molecules and then removing the atom indices that are a part of the core. The atom indices were found by first creating a molecule object from the SMILES string using the function *Chem.MolFromSmiles()* with the SMILES string as input. The atom indices were then retrieved by calling *GetAtoms()* on the molecule object.

- **Step 3: Convert atom index to token index**
  The conversion from atom to token indices was made by using the fact that all atom tokens (such as C and [Br]) contains one or more letters and all other SMILES tokens (such as bonds and control tokens used to describe ring structures) does not contain any letters at all. For each token in the SMILES string that did not contain a letter, all atoms after this token had their index increased by one. For each property token and start token the atom indices were also increased by one.

[1] https://www.rdkit.org/

[2] https://github.com/rdkit/mmpdb

[3] In RDKit there is also a function called *FindMCS()*, which can be used to find the maximum common substructure between two molecules. However, in some edge cases, this way of finding the substructure was not a reliable alternative. Sometimes the wrong atom was said to be a part of the common substructure. This approach was used in a first attempt, but was changed to the approach described in Step 1 when we noticed that it did not always work.



src	0	1	2	3	4	5
gen	0	1	2	3	4	5

src	6

gen	6

atom					0		1	2	3	4	5		6	
token	0	1	2	3	4	5	6	7	8	9	10	11	12	13
src	P1	P2	P2	^	C	1	C	C	C	C	C	1	Cl	$

atom	0		1	2	3	4	5		6	
token	0	1	2	3	4	5	6	7	8	9
gen	C	1	C	C	C	C	C	1	O	$

Figure 3.2: A step-by-step example for how the attention weights between X and the R-group of the generated molecule are obtained. X corresponds to selected tokens in each experiment. In the first step, the core of the source and generated molecule are matched. This matching is then used in step two, where the R-group in the generated molecule is identified. In order to be able to retrieve the right attention weights, the atom indices are converted to token indices in step three. In step four, the attention between X and each atom in the R-group in the generated molecule is retrieved from an attention head.

- **Step 4: Find attention between X and R-group gen**
  After the atom indices have been converted to token indices, the attention weights between any tokens of interest can easily be retrieved by simply using their indices in the attention head. In most of the experiments the attention

between the atoms in the R-group in the generated molecule, highlighted in Figure 3.2, and some part of the input were retrieved. Which attention weights that are retrieved in each experiment will be explained in their corresponding sections.

3.2.2 Overview of attention weights over input token categories

We first created an overview of the attention weights for different token categories. All input tokens were divided into four different categories: start, property, SMILES and end tokens. The start token is ^ and the end-token $. They are used to inform the model of the start and end of the SMILES string. For each token category, two different attention weights were retrieved. The first measured the maximum attention to all the tokens in the output, while the second measured the maximum attention to each atom in the R-group in the generated molecule. As the point of this experiment is to provide an overview, the *general dataset* was used.

The reasoning behind taking the maximum attention weight rather than the average weight is mainly that it is of more interest to know whether some token had a large impact in generating a part of the output rather than if many tokens had a small impact in generating the output. Finding patterns for which tokens; that contribute a lot to different parts of the output, will provide a clearer explanation for what parts of the input that affect the output in a certain direction.

3.2.3 The effect of input property tokens on the transformation

To answer the research question regarding if the property constraints are important for the transformation, the maximum attention weight between the property tokens and each atom in the R-group was computed. The procedure is the same as described in Section 3.2.1 and the last step in this description is visualized on the left side of Figure 3.3. As the goal with this procedure is to determine if the property constraints are important in general, the *general dataset* was used.

As references, we compared with both how much other parts contributed to the transformation and how much the property tokens contributed to reconstructing the core. In the comparison with how much other parts contributed to the transformation, the input token categories described in Section 3.2.2 were used. Specifically, the attention weights between the input categories and the R-group atoms were compared.

To measure how much the property tokens contributed to reconstructing the core, the attention weights between the property tokens and the core in the generated molecule were retrieved. This procedure was similar to the procedure described above for the R-group. The only difference is that instead of the atoms in the R-group, the attention weights between the property tokens and the core atoms were obtained.

Figure 3.3: Step four, for the retrieval of attention weights between the property tokens and the R-group atoms.

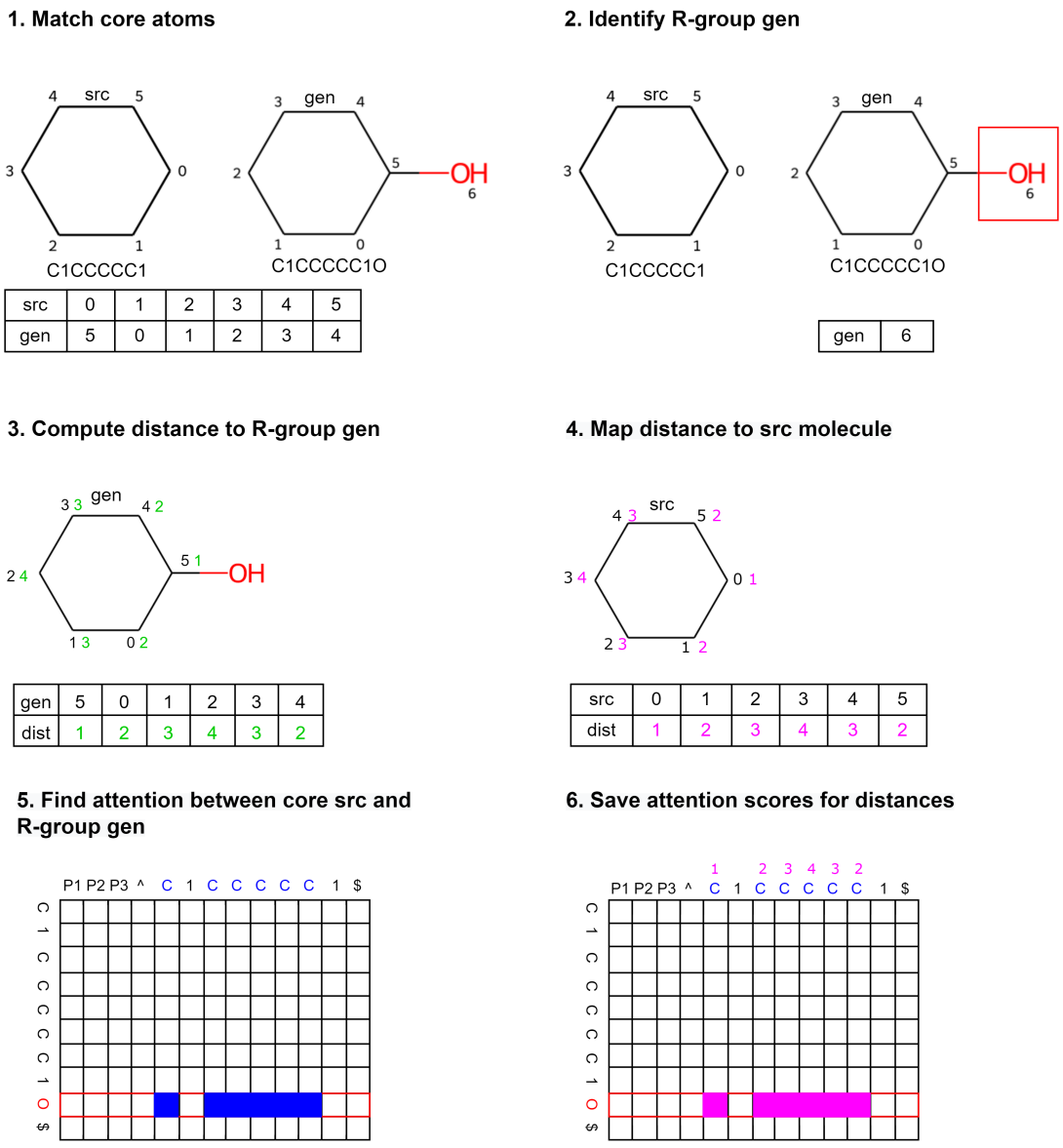
3.2.4 Relationship between topological distance and attention weights

In this section, we will describe the method used to investigate how atoms, depending on their placement in the source molecule, contribute to different part of the generated molecule. In particular, we are interested in if atoms close to each other in the source molecule will have higher attention weights between each other than atoms that are further away. This was investigated for the core atoms in the source molecule to the atoms in the R-group and core in the generated molecule individually. Also for this investigation the *general dataset* was used.

This method can be divided in three main parts. First, we retrieved the attention weights and the topological distances between all atoms of interest. This procedure is described in detail in Section 3.2.4.1. We then looked at the distribution of the maximum topological distance between any atoms in the R-group and core of the generated molecule, respectively (Section 3.2.4.2). These distances were then used to determine how many bins the atoms should be divided into in the next step. The last step was to divide the atoms into a number of bins according to their distance to the atom of interest, such that the atoms closest to the atom of interest were put in the first bin and the atoms the furthest away in the last bin. This procedure is described in Section 3.2.4.3.

3.2.4.1 Retrieval of attention weights and topological distances

The retrieval of attention weights and topological distances is illustrated step-by-step in Figure 3.4. In the first step, the common core of the source and generated molecule are matched to each other. Each atom in the core of the generated molecule is mapped to its corresponding atom in the source molecule. In the second step, this matching is used to identify the R-group in the generated molecule. The third step was to compute the distance between each atom in the R-group and the atoms in the core. This was done using the *GetDistanceMatrix()* in RDKit, which takes a molecule in SMILES format as input and returns a distance matrix containing



src	0	1	2	3	4	5
gen	5	0	1	2	3	4

gen	6

gen	5	0	1	2	3	4
dist	1	2	3	4	3	2

src	0	1	2	3	4	5
dist	1	2	3	4	3	2

Figure 3.4: A step-by-step example for how the attention weights and topological distances between the R-group of the generated molecule are obtained. In the first step, the core of the source and generated molecule are matched. This matching is then used in step two, where the R-group in the generated molecule is identified. In the third step, the distance to the atoms in the R-group is computed for each core atom. These distances are, in step four, mapped to the source molecule. In step five, the attention weights between the R-group atoms in the generated molecule and the core atoms in the source molecule are retrieved. In the last step, these attention weights are saved together with their topological distance to the R-group atom.

the distances between all atoms in the molecule. In the fourth step, the topological distances were mapped to the corresponding atoms in the source molecule, giving each core atom in the source molecule a distance to the R-group in the generated molecule. After this conversion, the mapping between the R-group in the generated molecule and the core in the source molecule allowed us to retrieve the attention weights between them. The last step was to save these attention weights with its corresponding topological distance to the R-group. For more details on step 1-2 and the conversion from atom indices to token indices, see Section 3.2.1.

3.2.4.2 Size distribution of the R-group and core in the generated molecule

As the molecules differ in number of atoms and shapes, only considering the absolute distances could be misleading. In smaller round molecules the distances between the atoms will be shorter and the in larger and straighter molecules the distances between the atoms will in general be longer, see Figure 3.5. In a larger and straighter molecule, it might be reasonable to consider a larger absolute distance as a short distances, than what could be in a smaller and rounder molecule.

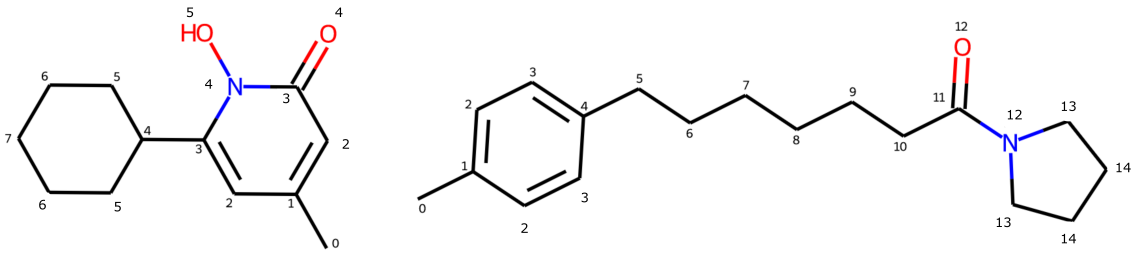


Figure 3.5: An example of two molecules that differ in number of atoms and shape. The SMILES string for the molecule to left is *Cc1cc(C2CCCCC2)n(O)c(=O)c1* and the SMILES string for molecule to the right is *Cc1ccc(CCCCCCC(=O)N2CCCC2)cc1*. The numbers are the topological distances from atom zero. Atom zero was chosen, so that the topological distance to the atom the furthest away from it would be the maximum topological distance in the molecule.

To take the size and shape of the molecules into account, we divided the distances into different bins according to how large the distance was compared to the other distances for that specific atom. To determine how many bins should be used, the distribution of the sizes of both the R-group and the core were used. Here we have defined the size as the maximal topological distance between any atoms in the specified subpart of the molecule, i.e., the size of the R-group is the maximal topological distance between any two atoms in the generated molecule that both belongs to the R-group.

A core atom in the source molecule with a high attention weight to an atom in the R-group in the generated molecule might contribute to generating the entire R-group, rather than just that specific atom. If this atom is close to some atoms in the R-group, but further away from the atom that it has a large attention weight to,

the result might therefore be misleading. This problem is a significant bigger issue if the size of the R-group is large in relation to the size of the core and increases with the number of bins.

The sizes of the R-group and the core were computed for all generated molecules in the *general dataset*. The generated molecules without any R-group were omitted in the size distribution result for the R-group, but not for the core.

A decision to only use two bins was decided after assessing the difference between the size distribution of the core and R-group of the generated molecule. The relative size of the R-group was determined to be too large to justify using more than two bins.

3.2.4.3 Dividing atoms into bins

For each atom in the R-group of the generated molecule, the topological distances to the core atoms were computed. The core atoms were then divided into two different bins, such that the half of the atoms with the shortest topological distance were put in the first bin and the other half of the atoms were put in the second bin. If the number of atoms in the core was odd, one more atom was put in the second bin.

The attention weight was then retrieved by matching the core in the generated molecule to the source molecule, as described in Section 3.2.4.1. For each atom in the R-group and bin, the maximal attention weight was retrieved and saved. All the saved attention weights were then visualized, in Figure 4.5, as a box plot for each bin. The same method was also used when measuring the attention weights between the core in the source and generated molecules. In this case, everything done for the R-group in the generated molecule was instead done for the core in the generated molecule.

3.2.5 The effect of the R-group of the source molecule on the transformation

The attention weight retrieval is already described in general terms in Section 3.2.1. In this experiment, the fourth step was to retrieve the attention weights between the R-group atoms of the source molecule and the R-group atoms in the generated molecule. Which attention weights this corresponds to is visualized on the left side of Figure 3.6. For each R-group atom of the generated molecule, both the average and maximum attention were saved.

In the right part of Figure 3.6 the attention weights between the core atoms of the source molecule and the R-group of the generated molecule were retrieved. Also, for these, the maximum and average attention weights for each R-group atom of the generated molecule were saved. These attention weights were used as a relative comparison for the attention weights between the R-groups and helped to determine if those could be considered to be high. As this procedure requires that both the source and generated molecule have an R-group, it was performed on the *R-group dataset*.

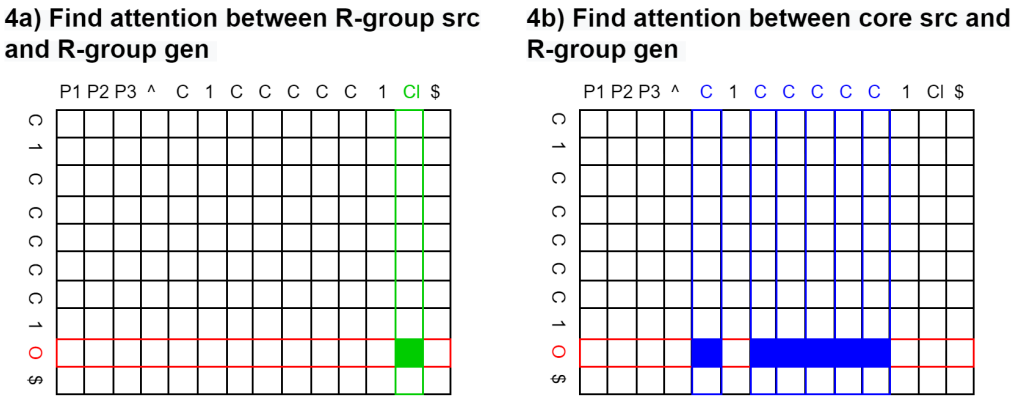


Figure 3.6: Step four, for the retrieval of attention weights between different atoms in the source molecule and the R-group atoms in the generated molecule. To the left, the attention weight between the R-group atom in the source and the R-group atom in the generated molecule is obtained. To the right, the attention weights between the core atoms of the source molecule and the R-group atom in the generated molecule are retrieved.

4

Results and discussion

In this chapter, we will present all our results, make some interesting observations and discuss their implications. We will begin by giving an overview of the attention weights for different input token categories. This overview will be given both for the maximum attention weights between each category and the atoms in the R-group, and for the maximum attention weights between each category and all the output tokens.

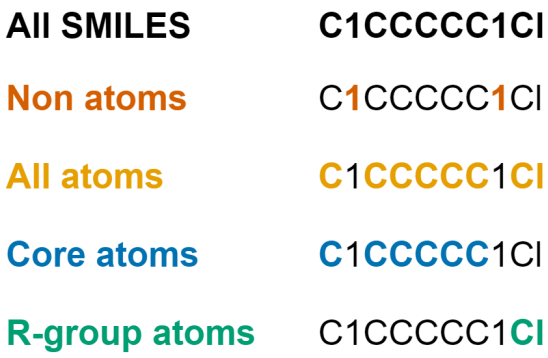


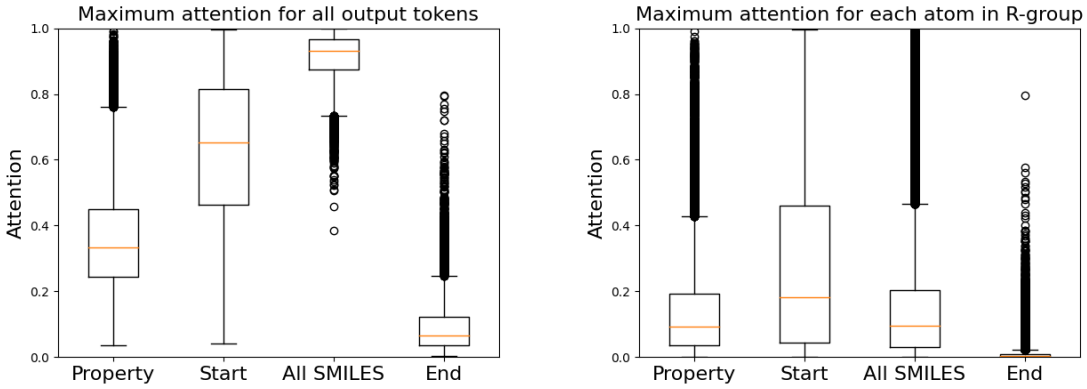
Figure 4.1: A visualization of which tokens that corresponds to each category.

After the overview, we will move on to presenting all the results for all research questions. Also in the results, the research questions were divided into the three categories: property change, molecular structure, and transformed part of the source molecule's effect on the transformation. In some plots, attention weights involving different subsets of the SMILES tokens will be used. To make it clear which tokens that belongs to which category, a visualization of this is showed in Figure 4.1. All the statistical presentations that are visualized were performed on attention head 1. The same visualizations for the other heads can be found in Appendix C. Moreover, the results for the other heads will be discussed briefly in the corresponding result section.

Last, we will discuss the high attention weight between the start token in the input sequence and the tokens in the generated sequence.

4.1 Overview of attention weights over input token categories

Here, we have presented an overview of how the maximum attention weights are distributed over four different input token categories. The first category is the property tokens: logD, solubility and clint. The second category is only the start token, which is used to show that the SMILES string begins. The third category is all the SMILES tokens, which describes the molecule. The fourth category is the end token, which constitutes the end of the SMILES string.



(a) Maximum attention weights for each token category and all output tokens.

(b) Maximum attention weights for each token category and atom in the R-group of the generated molecule

Figure 4.2: An overview of attention weights over the different input token categories: property change, start, SMILES, and end tokens. The main body is the data point within the quartiles (i.e., between 25:th percentile, Q_1 and 75:th percentile, Q_3). The vertical lines are the whiskers, here defined as all data points in the interval $[2.5Q_1 - 1.5Q_3, 2.5Q_3 - 1.5Q_1]$. These whisker parameters are commonly used in box plots and corresponds to the interquartile range (IQR) 1.5. The circles are the outlier data points, defined as all data points outside the whisker interval.

The results for each category are visualized with box plots in Figure 4.2. On the left side (Figure 4.2a), the maximum attention weight for each category is taken over all the output tokens. In other words, for each input-output pair, only one maximum attention weight is obtained for each input token category and molecule pair. On the right side of the Figure (Figure 4.2b), the maximum attention weight is instead obtained for each atom in the generated molecule that belongs to the R-group.

The same general trends were found for most of the heads. However, which of the start and SMILES token category that had the largest median attention weights, in both Figure 4.2a and 4.2b varied between the different heads. The same plots for all eight heads can be found in Figure C.1 and C.2 in Appendix C.

In Figure 4.2b the property and the SMILES token have similar maximum attention weights, while the the attention weights are larger for the start token and smaller for the end token. This indicates that the property and SMILES token that contribute most to the atom in the R-group are of equal importance. It also indicates that the start token contributes more to the generation of the R-group. As the start token only is used to tell the model that the SMILES string begins and looks the same for all inputs, it is unlikely that it should affect the generation of the R-group to this extent. A further discussion of this behaviour is presented in Section 4.5.

In Figure 4.2a the SMILES tokens have the highest median of the maximum attention weights, meaning that the maximum attention weight in the attention head often was a SMILES token. The large difference between the maximum attention weights of the SMILES tokens between Figure 4.2a and 4.2b can be explained by that the highest attention weights involving SMILES tokens were found between core atoms. This observation was first observed in the heatmaps of the attention heads (an example is shown in Figure B.1 and B.2 in Appendix B) and will be described in further detail in Section 4.3.2.

From Figure 4.2a, we can also see that the property and start token have significantly higher values compared to in Figure 4.2b. This indicates that they are important, not only for the transformation, but also for the recreation of the core of the molecule.

4.2 The effect of input property tokens on the transformation

To determine the relative importance that the input property tokens had on the transformations, two reference points were used. First, we compared with how much other input token categories contributed to the transformation. Then, we compared it with how important the property tokens were for regenerating the other atoms, i.e., recreating the core.

From Figure 4.2b, we can see that the maximum attention weight from the property tokens to an atom in the R-group on average is roughly the same as for the maximum attention weight between all SMILES tokens and an atom in the R-group. This means that the property token that contributes most to the generation has about the same contribution as the SMILES token that contributes the most. As there are many more SMILES tokens compared to property tokens, this supports the assumption that the property tokens are important for the transformation.

In Figure 4.3 the attention weight between the property tokens and the R-group is generally slightly higher than the attention weight between the property tokens and the core. The same trend could be seen for all other heads. See Figure C.3 in Appendix C for the same visualization for all eight heads.

The consistency of higher attention weights for the R-group supports the assumption that the property change should contribute more to the transformation than to the preservation of the core. However, as the difference in attention weights between the R-group and the core was small more investigation, such as using other explainability

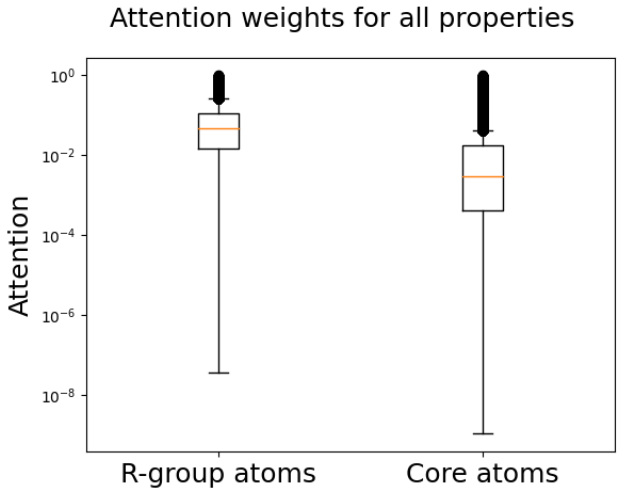


Figure 4.3: Box plots of the attention weights between the property tokens and R-group in the generated molecule and the core, respectively. The attention weights are computed for each atom in the R-group and core.

methods, could be useful to strengthen the argument. Due to time constraints we did not do this ourselves, but a brief discussion of another explainability methods will be presented under further work in Section 5.

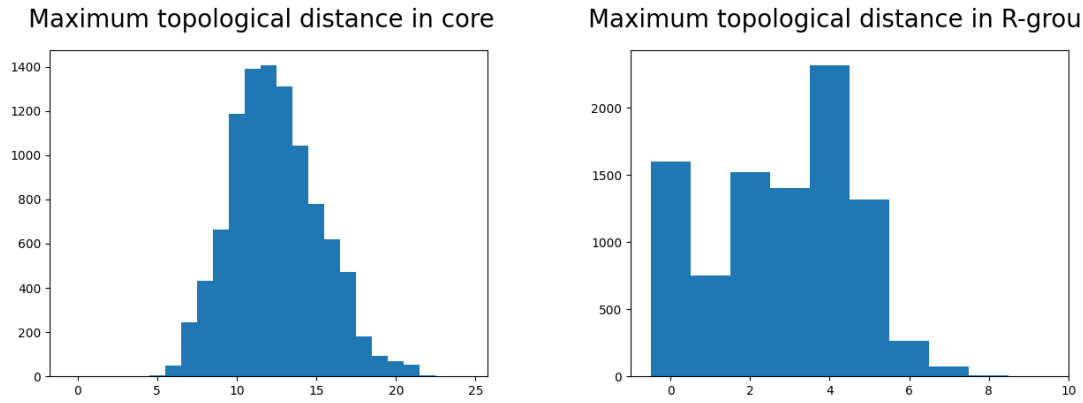
4.3 Relationship between topological distance and attention weights

Here, we have presented the results for how the topological distance between atoms affect the attention weights. First, we visualize the maximum topological distances of the core and the R-group in the generated molecule. These were presented first, as they were used to decide the number of bins that should be used for the topological distance analysis. Secondly, the results for how the topological distance between atoms affect the attention weights are presented. Last, a breakdown of the attention weights for the different SMILES tokens is shown. This breakdown was made to explain the large difference between the attention weights between atoms in the topological distance result and the attention weights of all SMILES tokens in the overview. For more information about how and why these results were obtained, see Section 3.2.3 in the method.

4.3.1 Size distribution of the R-group and core in the generated molecule

We here define the size as the maximum topological distance between all atoms.

Figure 4.4a shows the size distribution of the core of the generated molecule. The most common maximum distance within the core is 13 and the distribution looks similar to a normal distribution.



(a) Distribution of the maximal distance between any two atoms in the core of each molecule.

(b) Distribution of the maximal distance between any two atoms in the R-group of each generated molecule.

Figure 4.4: The distributions of maximum topological distance within the core and within the R-group, respectively.

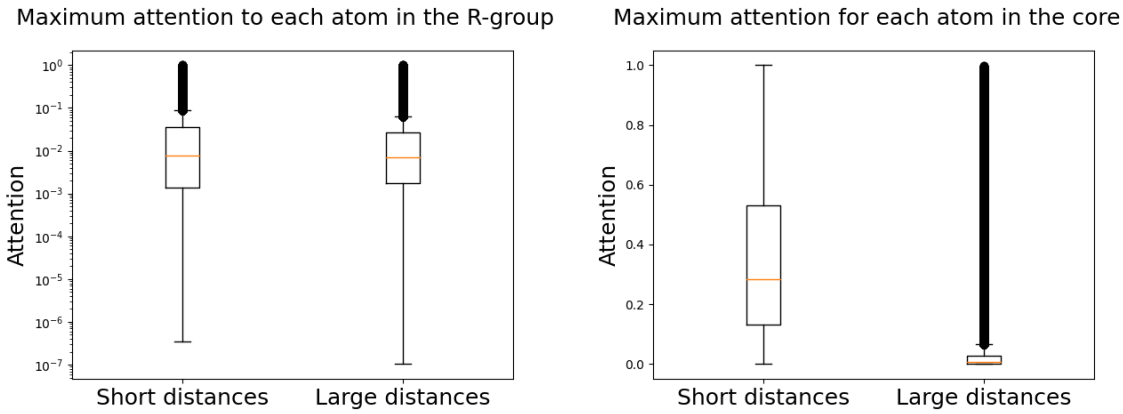
Figure 4.4b shows the size distribution of the generated molecule. Note that there are generated molecules without any R-group, which are not represented in this image. The most common maximal distance is distance 4 with over 2000 occurrences. Distances 0, 2, 3 and 5 all have roughly 1500 occurrences each. The zero distances means that there is only a single atom in the R-group of the generated molecule.

After comparing the size distributions of the R-groups and cores, it was decided to only use two bins in the topological distance analysis. This decision was made as the relative size of the R-group was too large to justify using more bins.

4.3.2 Attention weights for short and long distances

For this analysis, the atoms were sorted into different bins according to their distance to the atom of interest. From the distribution results of maximum topological distances within the core and the R-group, visualized in Figure 4.4, it was decided to only use two bins. In the first bin, the 50 % of the atoms with the shortest distances to the atom of interest were placed and in the second bin the rest of the atoms were placed. For more details about the procedure and the motivation behind it, see Section 3.2.4 in the method.

In Figure 4.5 the attention weights for the short and large distances are visualized. On the left side of the figure (Figure 4.5a), the distance is measured from all core atoms of the source molecule to all R-group atoms of the generated molecule. The maximum attention weight was then taken for each bin and atom in the R-group. The atoms with short distances and the large distances to the atom in the R-group both had very low and similar attention weights. The very low attention weights, and the similar values, were seen in all heads. However, which of the distance categories that had the slightly higher median attention weight, differed between



(a) Maximum attention weights between the core of the source molecule and each atom in the R-group of the generated molecule.

(b) Maximum attention weights between the core of the source molecule and each atom in the core of the generated molecule.

Figure 4.5: Maximum attention weights for each bin and atom of interest to all core atoms. The atom of interest is either an atom in the R-group in the generated molecule or an atom in the core.

the heads. For the same visualization as in Figure 4.5a for all heads, see Figure C.4 in Appendix C.

The small difference in the magnitude of the attention weights between the short and long distances, indicates that the distance to the R-group is irrelevant for how much it contributed to its generation. However, to strengthen this claim other explainability methods could be used to verify that the attribution to the R-group is about the same for atoms with short and long topological distances to it. As mentioned in the introduction, the chemists' intuition is also that the atoms closer to the R-group will contribute more to the transformation when the property change is small. Both the use of other explainability methods and to specifically investigate the relevance of the topological distance for molecule pairs with small property changes will be discussed as future work in the conclusion chapter.

In Figure 4.5b, the distance is measured from all atoms of the source molecule to each core atom of the generated molecule. The maximum attention weight was obtained for each bin and core atom. It is clearly visible that the atoms with shorter distances to the core atoms have substantially higher attention weights than the ones that are further away. The same trend was visible for all eight heads, see Figure C.5 in Appendix C for the same visualization for all heads.

That the distance would be important for preserving the core of the molecule in the prediction coincides with our observation of the attention heatmaps. An example of all attention heatmaps for a molecule pair is visualized as heatmaps in Figure B.1 and B.2 in Appendix B. In some of these heatmaps there is a diagonal of high attention weights where each atom in the core of the molecule has high attention weights to themselves. This indicates that the model attends most to the atom

in the source molecule that it is recreating. In some of the heatmaps, it was also seen that the molecules close to each other generally had higher attention weights between them. This implies that the model uses the atoms close by to understand where to place the atoms in the generated molecule.

4.3.3 Breakdown of attention weights to SMILES tokens

When the maximum attention weights between the core of the source molecule and the R-group of the generated molecule in Figure 4.5a were compared to the maximum attention weights between the SMILES tokens of the input and the R-group atoms of the generated molecule in Figure 4.2b a large gap was observed. The attention weights in Figure 4.5a were substantially lower than the ones in Figure 4.2b. As both of these plots measure the maximal attention between the R-group and all core atoms (in Figure 4.5a) or all SMILES tokens (in Figure 4.2b) this large difference in attention weights was surprising. As the core atoms are a subset of all SMILES tokens, it is obvious that the maximal attention weights between the SMILES tokens and the R-group is higher than the attention weights between the core atoms and the R-group atoms. However, it was surprising that the difference was so large.

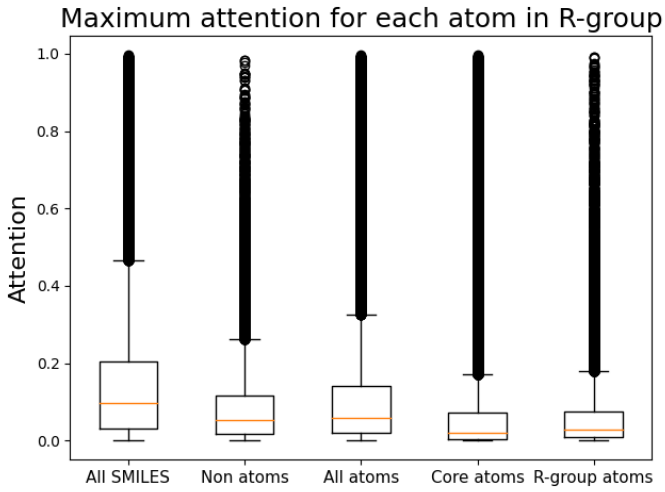


Figure 4.6: A breakdown of the maximum attention weights between the input SMILES tokens categories on the x-axis and the R-group in the generated molecule.

The previous assumption was that token pairs with atoms had higher attention weights between them than token pairs with other SMILES tokens. Therefore, the maximum attention weight between the core and the R-group atoms and between all SMILES tokens and the R-group atoms should be fairly similar. To explain this gap, a breakdown of the maximum attention weights for different SMILES tokens was performed.

Both between the categories All SMILES and all atoms and between all atoms and core atoms in Figure 4.6, there is a significant drop in the maximum attention weights. This means that the attention weights of the category atom token were

not as much higher compared to the other SMILES tokens, as we had expected. Moreover, as most of the atoms are core atoms, it was also surprising that the difference of their maximum attention was so large. The large differences between the categories explain the difference between Figure 4.5a and 4.2b.

The same result was found for most of the heads. However, for some heads, the attention weights for the all SMILES category were so small that the difference between the other categories were hard to distinguish. For the visualization in Figure 4.6, for all the heads, see Figure C.6 in Appendix C.

4.4 The effect of the R-group of the source molecule on the transformation

To determine the relative importance that the R-group in the source molecule had on the transformation, a reference point was used. This reference point is the contribution that the core atoms in the source molecule had to the transformation. For a detailed explanation of how the attention weights were retrieved, see Section 3.2.5 in the method.

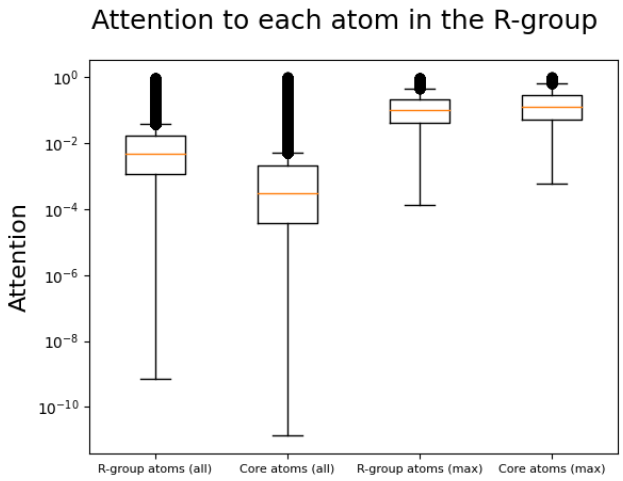


Figure 4.7: A visualization of the attention weights between the R-group in the generated molecule and the R-group in the source molecule and core, respectively. "All" in the Figure refers to all attention weights between the atoms in the R-group in the generated molecule. "Max" refers to the maximum value for each atom in the R-group in the generated molecule.

From Figure 4.7 in the first and second box plot, it is clear that the R-group in the source molecule in general have higher attention weights to the R-group in the generated molecule. This indicates that each atom in the R-group of the source molecule on average contributes more to the transformation than the atoms in the core of the source molecule.

In Figure 4.7 in the third and fourth box plot, we can also see that the maximum attention weights for the R-group and the core are roughly the same. This implies that the core atom and R-group atom, that contributed most to the transformation, had approximately the same magnitude of contribution. The same trends as have been discussed for Figure 4.7 were also seen for most of the heads. See Figure C.7 in Appendix C for the same plots for all heads.

4.5 Analysis of the start token

The results from the overview of the attention weights over input token categories (Section 4.1) showed that the start token had a relatively high attention weight, which was an unexpected result. However, a similar behaviour has also been seen in an analysis of the attention in the transformer based model BERT [4], where the BERT model focuses on the corresponding separator token called [SEP]. In the article, they propose that a possible explanation is that [SEP] could be used to aggregate segment-level information, which can then be read by other heads. To further investigate this hypothesis, they applied gradient-based measures of feature importance [35], where they compute the gradient of the loss from BERT's masked language modelling task with respect to each attention weight. Their test concluded that their initial hypothesis was wrong. The result instead showed that when the [SEP] becomes high, the gradients for attention to [SEP] becomes very small, denoting that attending more or less to [SEP] does not substantially change BERT's output and that the [SEP] act like an "no-op" (no operation). This implies that an attention head focuses on the [SEP] tokens when it can not find anything else in the input sentence to focus on.

Another example of this behaviour was shown by Kobayashi et al. [36] where they analysed the Transformer model using vector norms. They found that although the attention score of [SEP] token was high, the norm of the value vector that is being multiplied with the attention score was very low. So low that the final product ends up being close to zero. They argued that special tokens are considered as an operation that does not collect anything, just like a "no-op". In our case, the start token could act similar to a separator token such as [SEP], because it separates the property tokens and the SMILES tokens. Therefore, we believe that the start token in our case acts like a "no-op" as well and will not affect the output of the generation.

5

Conclusion

This thesis has focused on explaining the Transformer model [16] for molecular optimization presented in [3]. The aim was to investigate attention [22] as an explainability technique for chemical language, which has not yet been deeply explored. To do so, we have developed our own framework for analysing how different parts of the input affect the generated molecules. In particular, we wanted to investigate if the chemists' intuition regarding what part of the input is of most relevance when optimizing a molecule agrees with where the model puts its attention. This was investigated through our research questions: (i) Is the property change important for generating the transformation? (ii) Is the transformation affected most by the atoms closest to its corresponding part in the source molecule? (iii) Is the transformed part in the source molecule important for generating the transformation?

The main findings were: Firstly, the property tokens seem to have an important contribution to the transformation. Secondly, the R-group of the source molecule on average contributes more to the transformation than the atoms in the core of the source molecule. Finally, the chemists' intuition regarding the distance to the R-group being relevant for the contribution of the transformation does not seem to agree with how the model learns. Nevertheless, it is difficult to tell whether the distance to the R-group is truly irrelevant for the model or if simply looking at the attention weights is not a good method to investigate this particular dependency. Additionally, the Transformer's attention focuses a lot on the start token, similar to BERT attending to the separator token [SEP] [4, 36]. This is a result of the token acting as a "no-op" and does not affect the output.

The Transformer is, as all deep learning models, a black box, meaning that it is difficult to understand why a certain prediction is made. However, to be able to improve the model, it can be of great importance to understand what is most important for the predictions. Therefore, being able to explain so that human users can understand the results of a solution generated by Artificial Intelligence is of high relevance. In our specific case, where the Transformer has been used for molecular optimization, it is important to understand if the model shares the chemists' intuition for what transformations that are suitable. In general, the results of our experiments showed a resemblance with the chemists' intuition for which part of the input that will affect the transformation. However, the absolute attention weights, for both the input part that was believed to have a high impact and the part used for comparison, were generally low. Moreover, only three assumptions inspired by

the chemists' intuition were tested. To draw conclusions regarding how similar the chemists' intuition and the model's decision process truly are, further investigations are needed, e.g., doing similar analysis with other explainability techniques and test more assumptions based on the chemists' intuition.

Future work

In our thesis, we investigated the relationship of the topological distance between the core atoms and R-group and the core atoms' contribution to the R-group of the generated molecule. It was concluded that the distance did not seem to affect the contribution of the atoms at all. However, the chemists' intuition regarding the effect of the distance was also that it should be more important for small property changes. Due to time constraint, we did not investigate whether this seemed to be the case. However, this could easily by done creating one dataset with only large property changes and one with only small property changes. The same box plots as in Figure 4.5a for each dataset could then be made and compared. If the median of the maximum attention weight is higher for the dataset with small property changes, it will support the chemists' intuition.

The topological distance between the core atoms of the source and the core atoms of the generated molecule seemed to be of great importance for the reconstruction of the core. From our results, it is unfortunately not possible to determine if it is truly the topological distance rather than just the distance between the SMILES tokens that is important for the conservation. The distinction was not relevant for our research questions, however it might be interesting to look at. If it is actually the topological distance and not the distance between the SMILES tokens that is of most importance for the conservation of the core, it would imply that the model in some sense understands the molecular structure of the molecule. In other words, the model could understand how the SMILES strings are used to determine the graph-like structure of the molecule. To investigate if it is truly the topological distance rather than the distance between the SMILES tokens that matters, the effect of the different distances could be compared. This could be done similarly to our experiments for the topological distance; by using a bin for short distances and another for long distances. If the attention weights for the atoms with short topological distances are higher than those for the atoms with short distances between the SMILES token, it is reasonable to conclude that the model understand the topological distances.

To validate our findings, other explainability methods can be used. In particular, integrated gradient has shown potential as an explainability method. Unlike other gradient based methods, the integrated gradient interpolates between an empty input and the real input, and measures the effect each new part added to the input has in the output in every step. This interpolation means that integrated gradient does not break sensitivity. To satisfy sensitivity is defined as: for every interpolation of the input which differs in one feature, but yields different predictions, the differing feature most have a non-zero attribution [35]. Integrated gradient has been used in various fields of machine learning, such as image recognition, sentiment analysis and also in machine translation in the article "Towards Understanding Neural Machine

Translation with Word Importance" by He et al.[37] where it outperformed attention. Originally, our plan was to use the integrated gradient method ourselves, but due to time constraints we decided to omit it. However, it is fairly easy to implement and also yields a matrix with an attribution score for every input-output token pair. It therefore seems like a natural next step to use the integrated gradient to validate our findings.

Bibliography

[1] J. P. Hughes, S. Rees, S. B. Kalindjian, and K. L. Philpott, "Principles of early drug discovery," *British journal of pharmacology*, vol. 162, no. 6, pp. 1239–1249, 2011.

[2] N. Schaduangrat, S. Lampa, S. Simeon, M. P. Gleeson, O. Spjuth, and C. Nantasenamat, "Towards reproducible computational drug discovery," *Journal of cheminformatics*, vol. 12, no. 1, pp. 1–30, 2020.

[3] J. He, H. You, E. Sandström, E. Nittinger, E. J. Bjerrum, C. Tyrchan, W. Czechtizky, and O. Engkvist, "Molecular optimization by capturing chemist's intuition using deep neural networks," *Journal of cheminformatics*, vol. 13, no. 1, pp. 1–17, 2021.

[4] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does bert look at? an analysis of bert's attention," *arXiv preprint arXiv:1906.04341*, 2019.

[5] R. Johansson, "Lecture notes in nlp course," Oct 2021.

[6] W. L. Jorgensen, "The many roles of computation in drug discovery," *Science*, vol. 303, no. 5665, pp. 1813–1818, 2004.

[7] "Nci dictionary of cancer terms," last accessed 24 May 2022. [Online]. Available: https://www.cancer.gov/publications/dictionaries/cancer-terms/def/lead-compound

[8] L. Guan, H. Yang, Y. Cai, L. Sun, P. Di, W. Li, G. Liu, and Y. Tang, "Admet-score–a comprehensive scoring function for evaluation of chemical drug-likeness," *Medchemcomm*, vol. 10, no. 1, pp. 148–157, 2019.

[9] L. Di and E. Kerns, *Drug-like properties: concepts, structure design and methods from ADME to toxicity optimization*. Academic press, 2015.

[10] R. Irwin, S. Dimitriadis, J. He, and E. J. Bjerrum, "Chemformer: a pre-trained transformer for computational chemistry," *Machine Learning: Science and Technology*, vol. 3, no. 1, p. 015022, 2022.

[11] P. Schwaller, T. Laino, T. Gaudin, P. Bolgar, C. A. Hunter, C. Bekas, and A. A. Lee, "Molecular transformer: a model for uncertainty-calibrated chemical reaction prediction," *ACS central science*, vol. 5, no. 9, pp. 1572–1583, 2019.

[12] D. Weininger, "Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules," *Journal of chemical information and computer sciences*, vol. 28, no. 1, pp. 31–36, 1988.

[13] AstraZeneca. Advancing data and artificial intelligence. [Online]. Available: https://www.astrazeneca.com/sustainability/ethics-and-transparency/data-and-ai-ethics.html

[14] M. Danilevsky, K. Qian, R. Aharonov, Y. Katsis, B. Kawas, and P. Sen, "A survey of the state of explainable ai for natural language processing," *arXiv preprint arXiv:2010.00711*, 2020.

[15] E. D. Liddy, "Natural language processing," 2001.

[16] 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*, 2017, pp. 5998–6008.

[17] R. Winter, F. Montanari, A. Steffen, H. Briem, F. Noé, and D.-A. Clevert, "Efficient multi-objective molecular optimization in a continuous latent space," *Chemical science*, vol. 10, no. 34, pp. 8016–8024, 2019.

[18] S. Y. Fei-Fei Li, Justin Johnson, "Lecture notes in recurrent neural networks," Stanford University - Computer Science Department, May 2017.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[21] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[22] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[23] A. Galassi, M. Lippi, and P. Torroni, "Attention in natural language processing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 10, pp. 4291–4308, 2020.

[24] C. Gunaratna, "Drug metabolism & pharmacokinetics in drug discovery: a primer for bioanalytical chemists, part i," *Current separations*, vol. 19, no. 1, pp. 17–24, 2000.

[25] Y. Kwon, *Handbook of essential pharmacokinetics, pharmacodynamics and drug metabolism for industrial scientists″*. Springer US, 2002, pp. 44–45.

[26] K. T. Savjani, A. K. Gajjar, and J. K. Savjani, "Drug solubility: importance and enhancement techniques," *International Scholarly Research Notices*, vol. 2012, 2012.

[27] J. Hussain and C. Rea, "Computationally efficient algorithm to identify matched molecular pairs (mmps) in large data sets," *Journal of chemical information and modeling*, vol. 50, no. 3, pp. 339–348, 2010.

[28] "Chembl database," EBI, last accessed 24 May 2022. [Online]. Available: https://www.ebi.ac.uk/chembl/

[29] S. M. Mathews, "Explainable artificial intelligence applications in nlp, biomedical, and malware classification: a literature review," in *Intelligent computing-proceedings of the computing conference*. Springer, 2019, pp. 1269–1292.

[30] J. Mullenbach, S. Wiegreffe, J. Duke, J. Sun, and J. Eisenstein, "Explainable prediction of medical codes from clinical text," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 1101–1111. [Online]. Available: https://aclanthology.org/N18-1100

[31] R. Ghaeini, X. Fern, and P. Tadepalli, "Interpreting recurrent and attention-based neural models: a case study on natural language inference," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 4952–4957. [Online]. Available: https://aclanthology.org/D18-1537

[32] J. Bastings and K. Filippova, "The elephant in the interpretability room: Why use attention as explanation when we have saliency methods?" *arXiv preprint arXiv:2010.05607*, 2020.

[33] S. Jain and B. C. Wallace, "Attention is not explanation," *arXiv preprint arXiv:1902.10186*, 2019.

[34] S. Wiegreffe and Y. Pinter, "Attention is not not explanation," *arXiv preprint arXiv:1908.04626*, 2019.

[35] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3319–3328.

[36] G. Kobayashi, T. Kuribayashi, S. Yokoi, and K. Inui, "Attention is not only a weight: Analyzing transformers with vector norms," *arXiv preprint arXiv:2004.10102*, 2020.

[37] S. He, Z. Tu, X. Wang, L. Wang, M. R. Lyu, and S. Shi, "Towards understanding neural machine translation with word importance," *arXiv preprint arXiv:1909.00326*, 2019.

A

Sample from the full dataset

A sample from the *full dataset*, with columns: source molecule, predicted molecule, transformation, core, delta logD, delta solubility, delta clint, source molecule's solubility, source molecule's logD, source molecule's clint, predicted molecule's solubility, predicted molecule's logD, predicted molecule's clint, are shown in Figure A.1.

	Source_Mol	Predicted_Mol	Transformation	Core	Delta_LogD	Delta_Solubility
0	O=C(OC1C[N+]2(CCCc3cccc(C(F)(F)F)c3)CCC1CC2)C(...	O=C(OC1C[N+]2(CCCOc3ccccc3)CCC1CC2)C(O)(c1cccc...	[*:1]c1cccc(C(F)(F)F)c1>>[*:1]Oc1ccccc1	[*:1]CCC[N+]12CCC(CC1)C(OC(=O)C(O)(c1ccccc1)c1...	(-0.7, -0.5]	no_change

Delta_Solubility	Delta_Clint	Source_Mol_Solubility	Source_Mol_LogD	Source_Mol_Clint	Predict_smi_cSolubility	Predict_smi_cLogD	Predict_smi_cClint
no_change	no_change	2.028984	0.837012	1.324445	2.336483	0.09996	1.435257

Figure A.1: A sample from the *full dataset*.

B

Visualization of attention heatmaps

Example of eight attention heads for transforming the source molecule's SMILES *'Cc1nc(N)nc2c1c(-c1ccc3oc(N)nc3c1)nn2C(C)C'* into the generated molecule's SMILES *'CNc1nc2c(c(-c3ccc4oc(N)nc4c3)nn2C(C)C)c(C)n1'*. The horizontal axis label shows each token of the three property changes concatenated with the source molecule's SMILES tokens. The vertical axis label shows each token of the generated molecule's SMILES tokens. The red highlighted bar represents the R-group in the generated molecule. In both axis labels there are two indexings. The index closest to the token is the regular index starting from *0* to *n*, *n* being the length of the molecule. The second index 'd_' represents the topological distance from the current atom to the R-group.

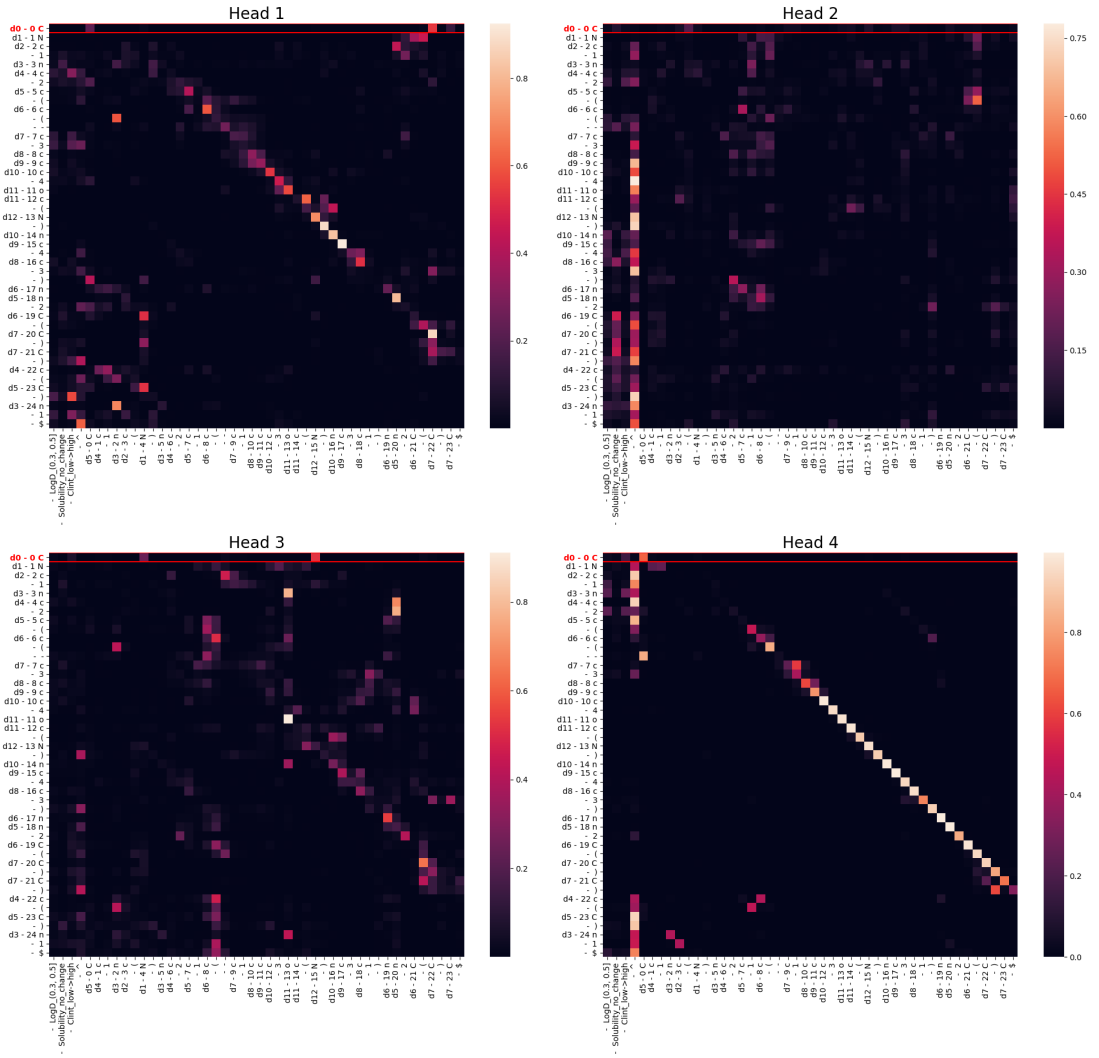


Figure B.1: Heatmaps of attention heads 1-4.

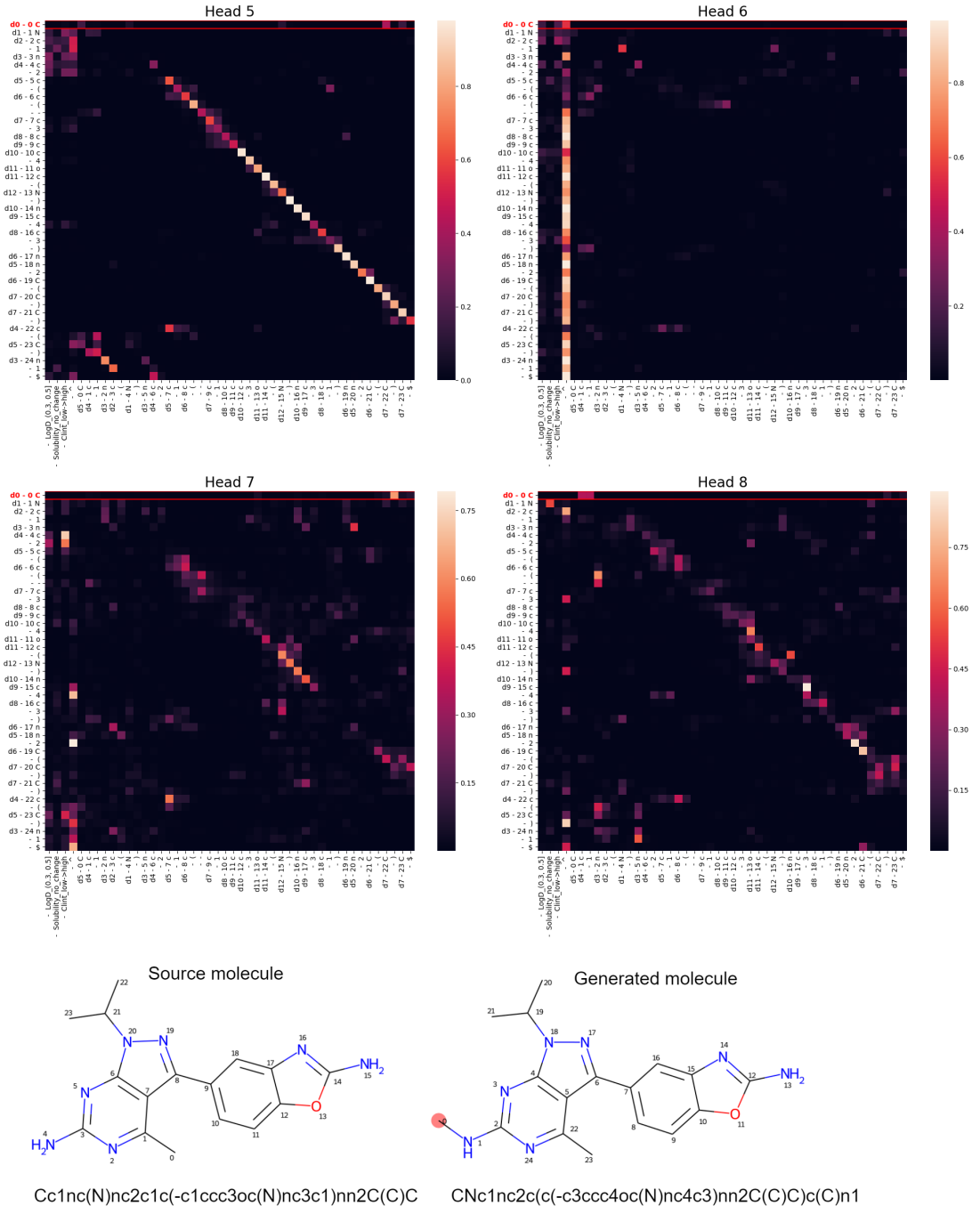
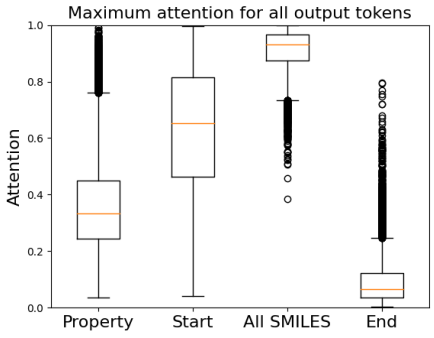


Figure B.2: Heatmaps of attention heads 5-8, and figures of the source and generated molecule with corresponding SMILES string.

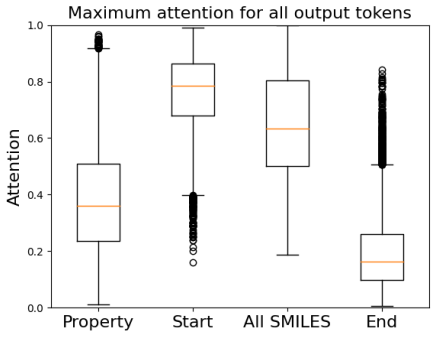
C

Additional results for all heads

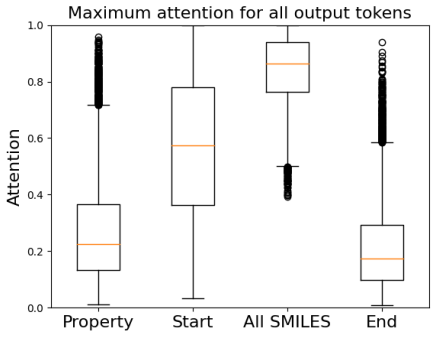
In this appendix, all heads will be presented for the result visualized in Section 4.1-4.4 in the result chapter. We will begin with the overview plots and then present the plots associated with each research question in order.



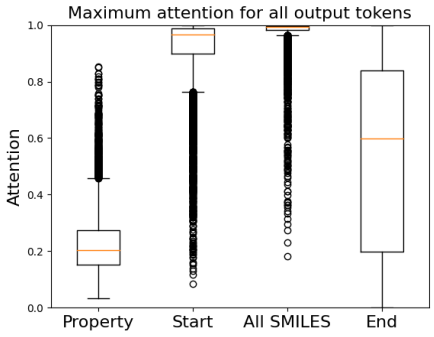
(a) Attention head 1



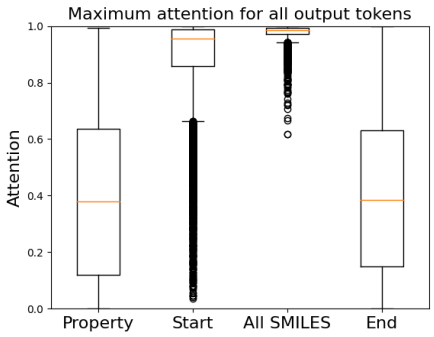
(b) Attention head 2



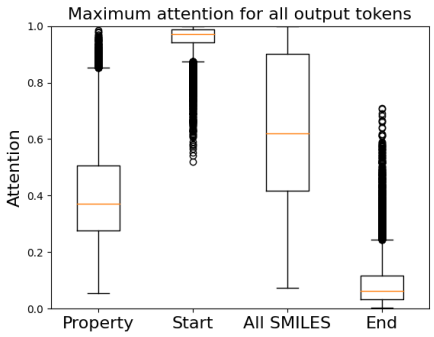
(c) Attention head 3



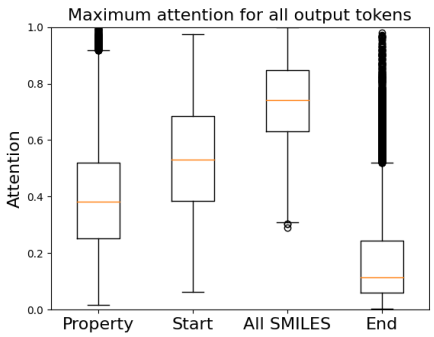
(d) Attention head 4



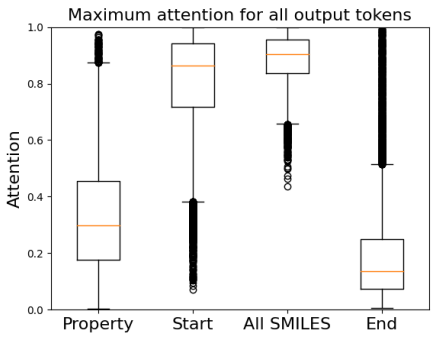
(e) Attention head 5



(f) Attention head 6

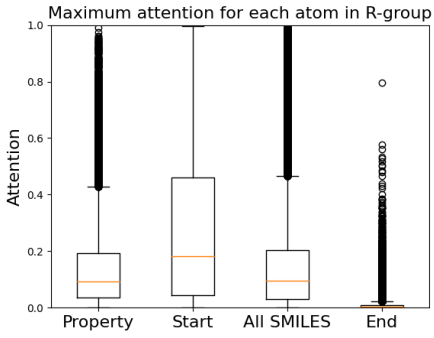


(g) Attention head 7

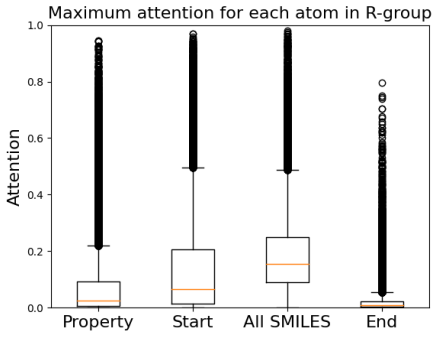


(h) Attention head 8

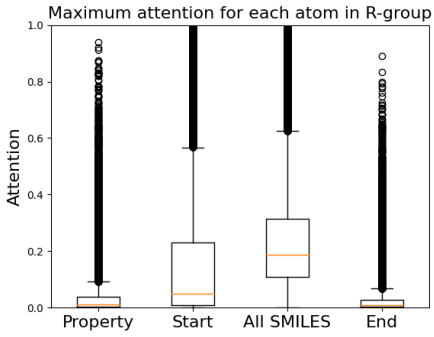
Figure C.1: An overview of the maximum attention weights between all output tokens and different input token categories: property change, start, SMILES, and end tokens.



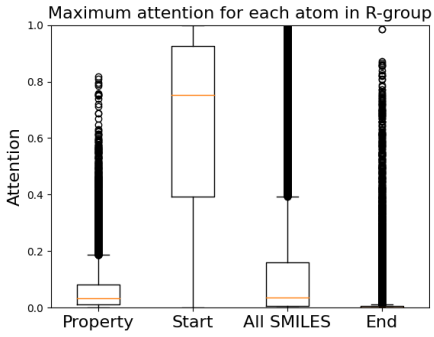
(a) Attention head 1



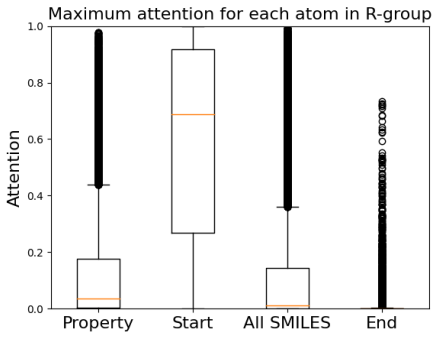
(b) Attention head 2



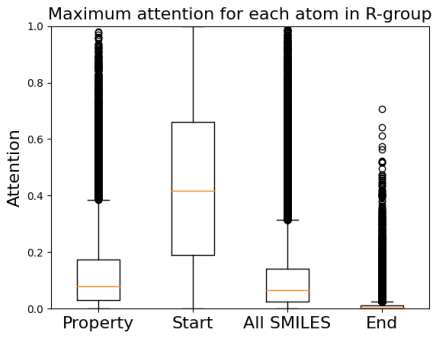
(c) Attention head 3



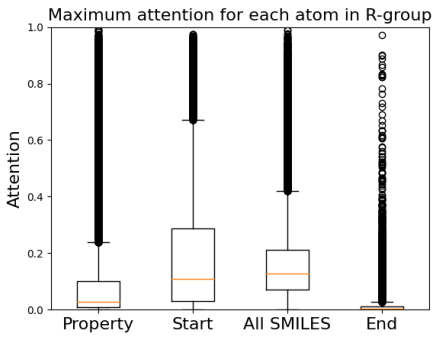
(d) Attention head 4



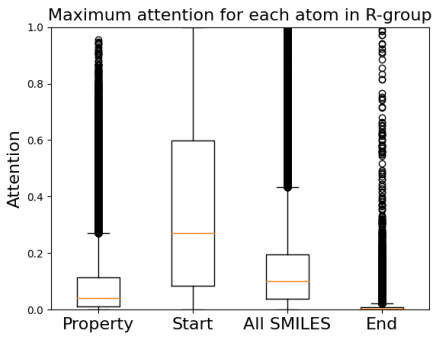
(e) Attention head 5



(f) Attention head 6

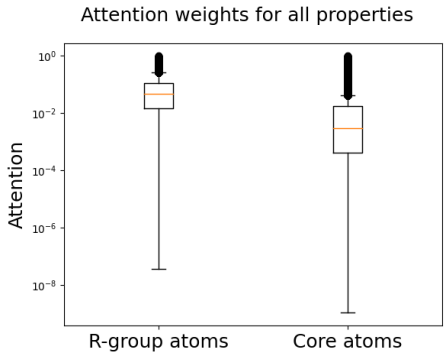


(g) Attention head 7

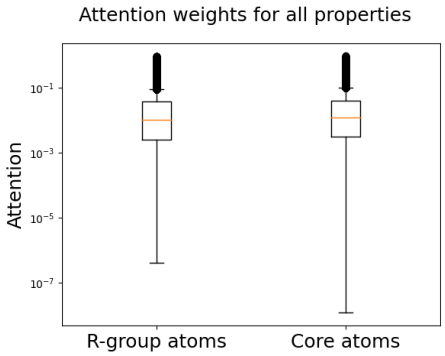


(h) Attention head 8

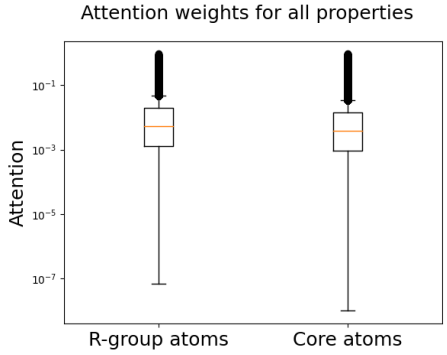
Figure C.2: An overview of the maximum attention weights between each atom in the R-group of the generated molecule and the different input token categories: property change, start, SMILES, and end tokens.



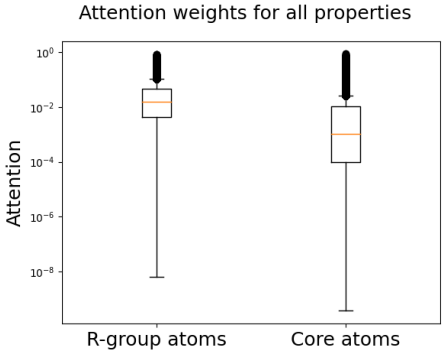
(a) Attention head 1



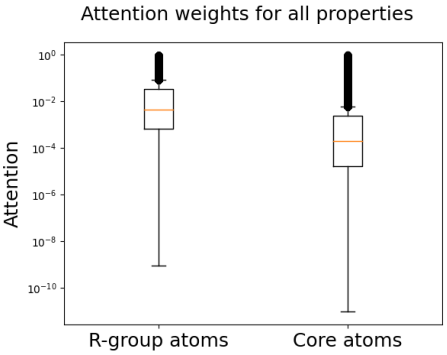
(b) Attention head 2



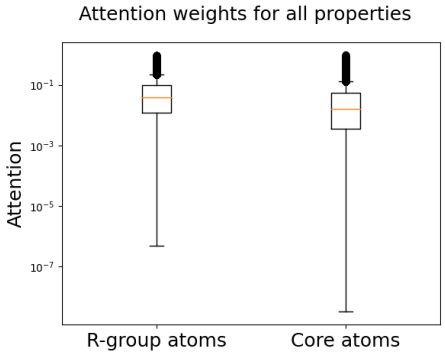
(c) Attention head 3



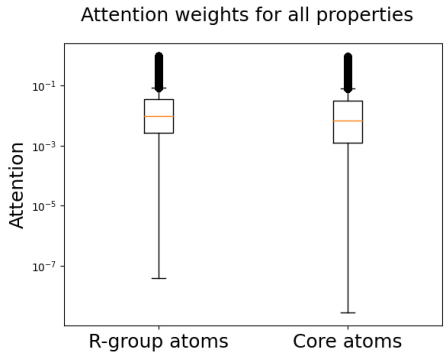
(d) Attention head 4



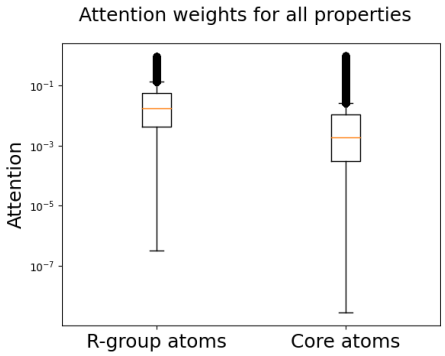
(e) Attention head 5



(f) Attention head 6

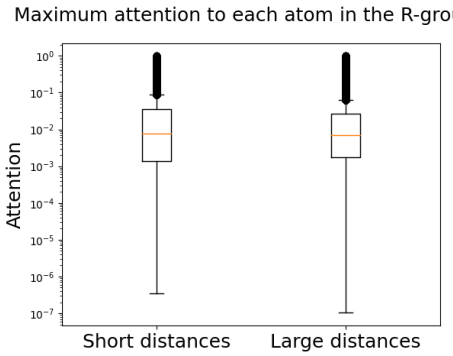


(g) Attention head 7

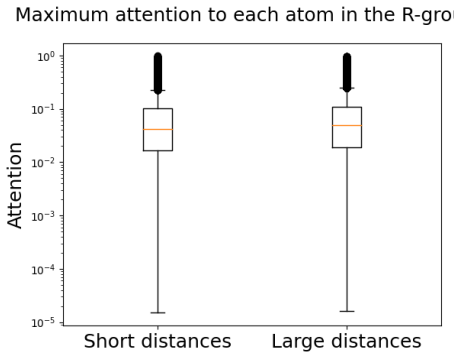


(h) Attention head 8

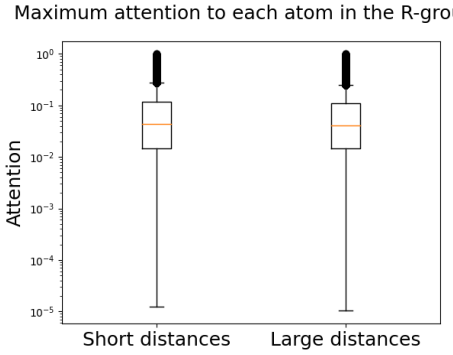
Figure C.3: Box plots of the attention weights between the property tokens and R-group in the generated molecule and the core, respectively. The attention weights are computed for each atom in the R-group and core.



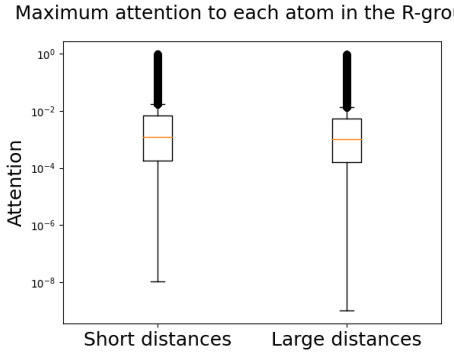
(a) Attention head 1



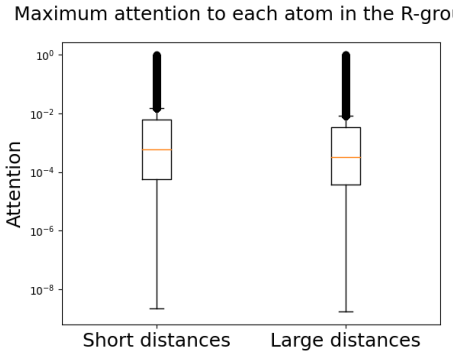
(b) Attention head 2



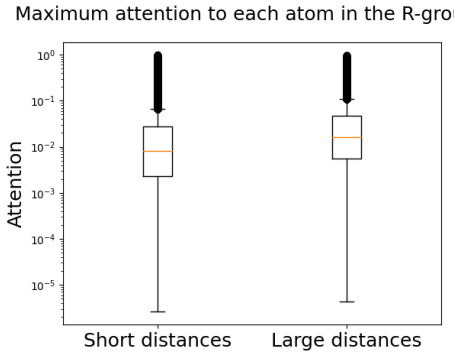
(c) Attention head 3



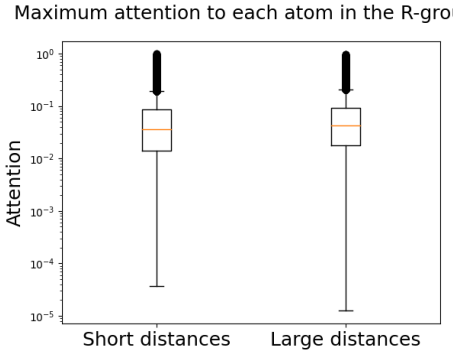
(d) Attention head 4



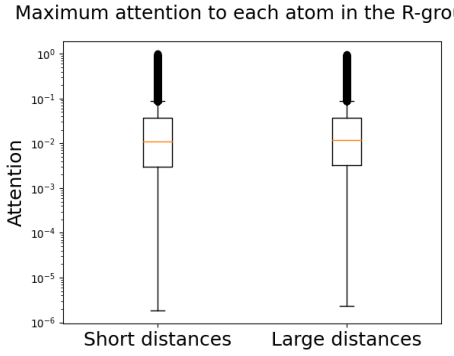
(e) Attention head 5



(f) Attention head 6

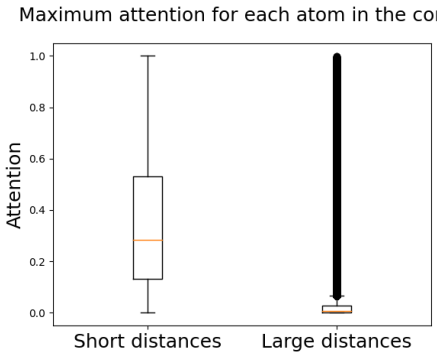


(g) Attention head 7

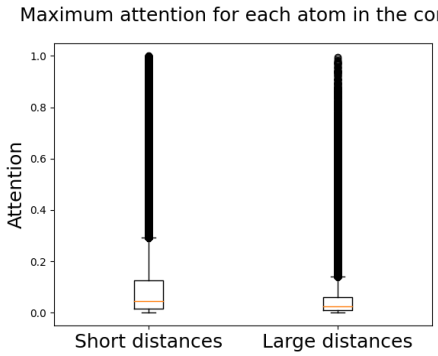


(h) Attention head 8

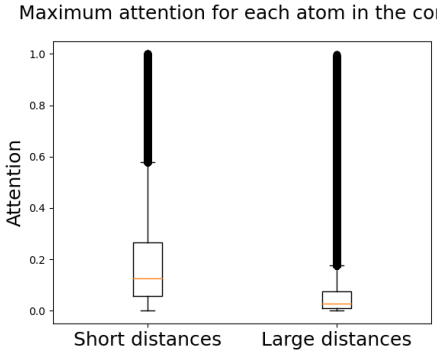
Figure C.4: Maximum attention weights for each bin and atom in the R-group of the generated molecule to all core atoms.



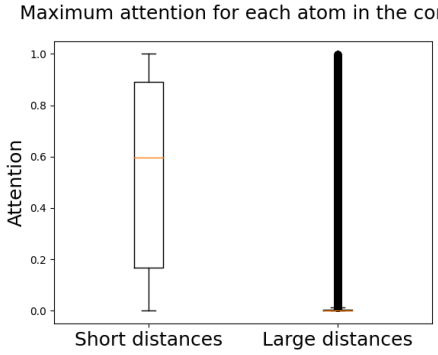
(a) Attention head 1



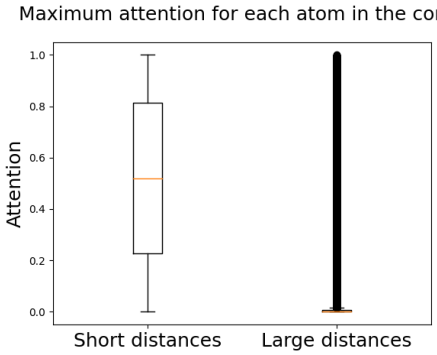
(b) Attention head 2



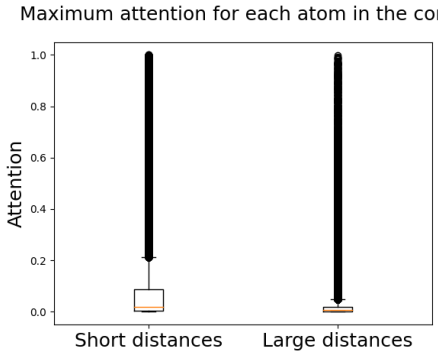
(c) Attention head 3



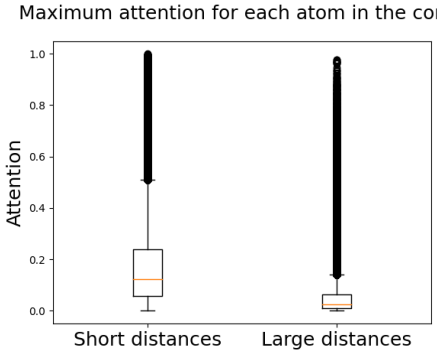
(d) Attention head 4



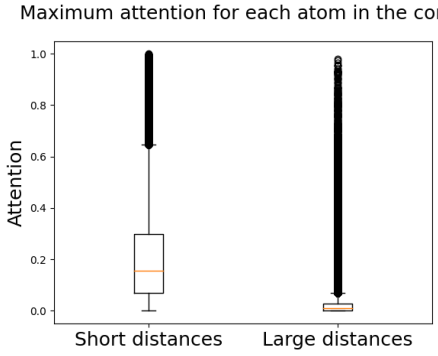
(e) Attention head 5



(f) Attention head 6

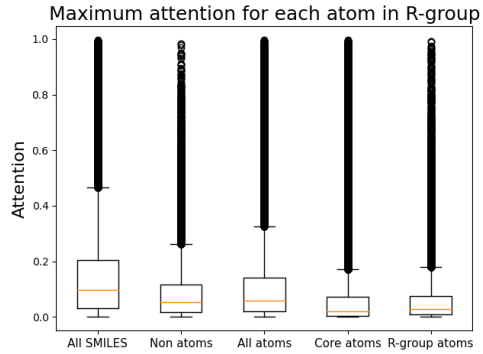


(g) Attention head 7

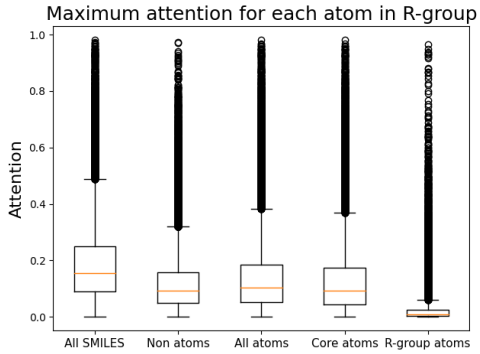


(h) Attention head 8

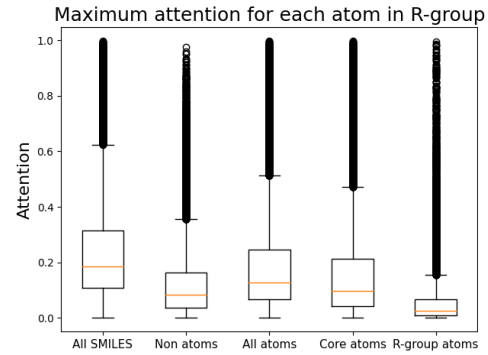
Figure C.5: Maximum attention weights for each bin and atom in the core of the generated molecule to all core atoms in the source molecule.



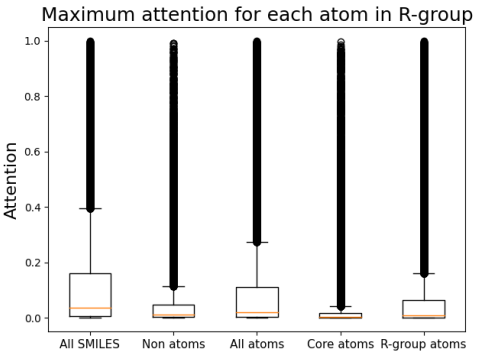
(a) Attention head 1



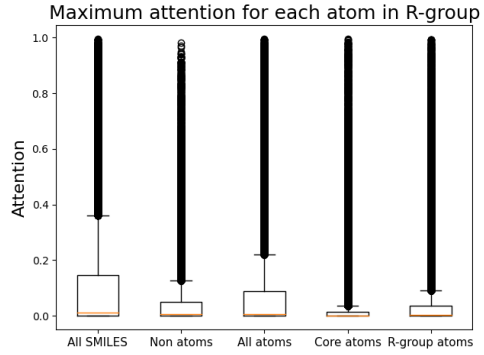
(b) Attention head 2



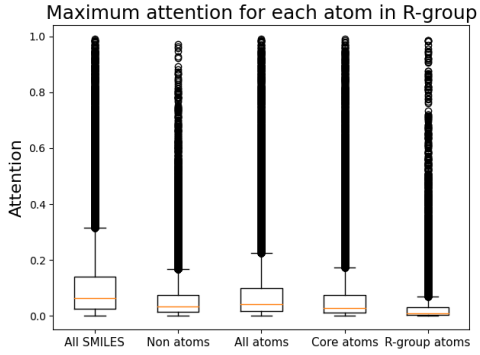
(c) Attention head 3



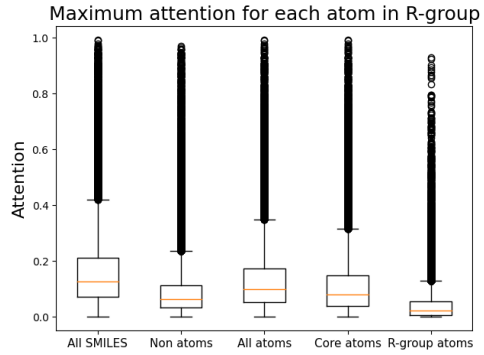
(d) Attention head 4



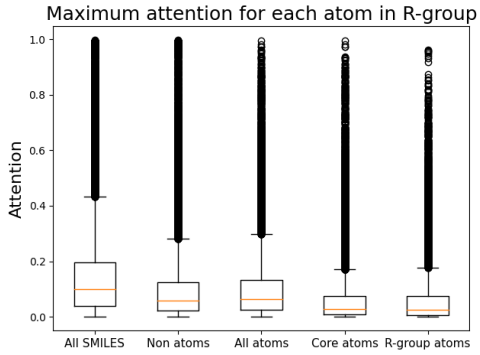
(e) Attention head 5



(f) Attention head 6

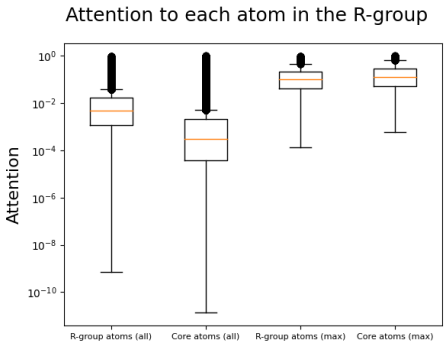


(g) Attention head 7

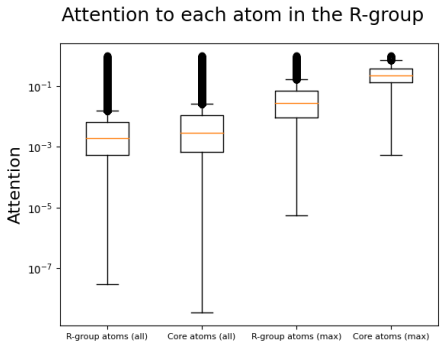


(h) Attention head 8

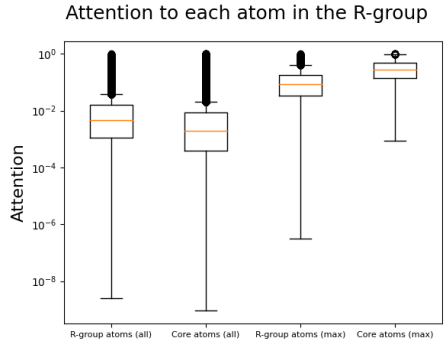
Figure C.6: A breakdown of the maximum attention weights between the R-group in the generated molecule and the different categories of SMILES tokens on the x-axis.



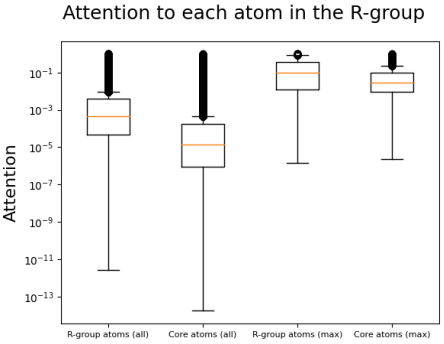
(a) Attention head 1



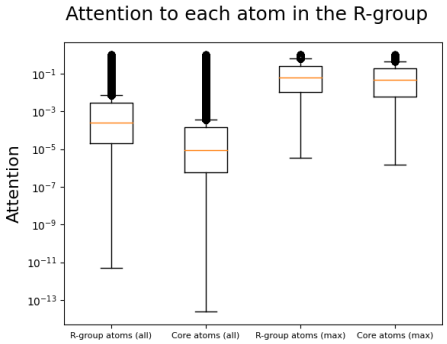
(b) Attention head 2



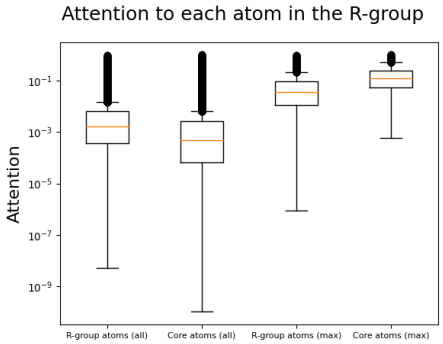
(c) Attention head 3



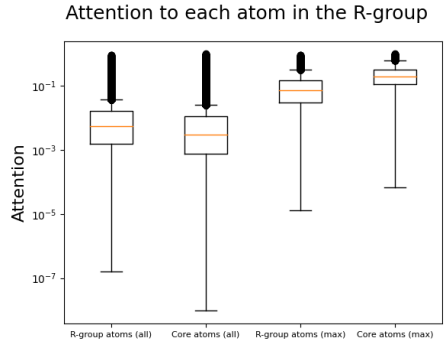
(d) Attention head 4



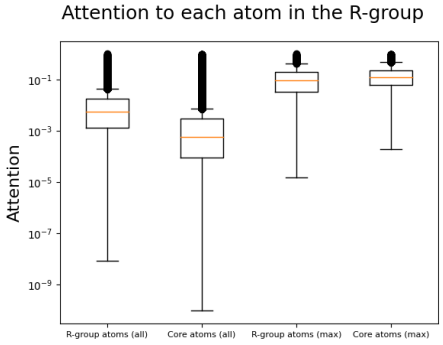
(e) Attention head 5



(f) Attention head 6



(g) Attention head 7



(h) Attention head 8

Figure C.7: A visualization of the attention weights between the R-group in the generated molecule and the R-group in the source molecule and core, respectively. "All" in the Figure refers to all attention weights between the atoms in the R-group in the generated molecule. "Max" refers to the maximum value for each atom in the R-group in the generated molecule.