# Analysis of Attention Mechanisms in Box-Embedding Systems

Jeffrey Sardina Jeffrey Sardina
*Trinity College Dublin, Dublin, Ireland*

Callie Sardina
*Boston College, Newton, MA, 02467, USA*

John Kelleher
*Technological University Dublin*, john.kelleher@tudublin.ie

*See next page for additional authors*

## Recommended Citation

Jeffrey Sardina, Jeffrey Sardina; Sardina, Callie; Kelleher, John; and O'Sullivan, Declan, "Analysis of Attention Mechanisms in Box-Embedding Systems" (2023). *Conference papers*. 38.
https://arrow.tudublin.ie/schfsehcon/38

Funder: Science Foundation Ireland D-REAL CRT under Grant Agreement No. 18/CRT6225 at the ADAPT SFI Research Centre at Trinity College Dublin, together with sponsorship of Sonas Innovation Ireland. The ADAPT SFI Centre for Digital Content Technology is funded by Science Foundation Ireland through the SFI Research Centres Programme and is co-funded under the European Regional Development Fund (ERDF) through Grant # 13/RC/2106_P2.

# Analysis of Attention Mechanisms in Box-Embedding Systems

Jeffrey Sardina[1](✉), Callie Sardina[2], John D. Kelleher[3], and Declan O'Sullivan[1]

[1] Trinity College Dublin, Dublin, Ireland
{sardinaj,Declan.OSullivan}@tcd.ie
[2] Boston College, Newton, MA 02467, USA
sardinac@bc.edu
[3] Technological University Dublin, Dublin, Ireland
john.d.kelleher@tudublin.ie

**Abstract.** Large-scale Knowledge Graphs (KGs) have recently gained considerable research attention for their ability to model the inter- and intra- relationships of data. However, the huge scale of KGs has necessitated the use of querying methods to facilitate human use. Question Answering (QA) systems have shown much promise in breaking down this human-machine barrier. A recent QA model that achieved state-of-the-art performance, Query2box, modelled queries on a KG using box embeddings with an attention mechanism backend to compute the intersections of boxes for query resolution. In this paper, we introduce a new model, Query2Geom, which replaces the Query2box attention mechanism with a novel, exact geometric calculation. Our findings show that Query2Geom generally matches the performance of Query2box while having many fewer parameters. Our analysis of the two models leads us to formally describe the interaction between knowledge graph data and box embeddings with the concepts of semantic-geometric alignment and mismatch. We create the Attention Deviation Metric as a measure of how well the geometry of box embeddings captures the semantics of a knowledge graph, and apply it to explain the difference in performance between Query2box and Query2Geom. We conclude that Query2box's attention mechanism operates using "latent intersections" that attend to the semantic properties in embeddings not expressed in box geometry, acting as a limit on model interpretability. Finally, we generalise our results and propose that semantic-geometric mismatch is a more general property of attention mechanisms, and provide future directions on how to formally model the interaction between attention and latent semantics.

**Keywords:** Box embeddings · Knowledge graph · Question answering · Attention

## 1 Introduction

This section is structured as follows: in Sect. 1.1, an introduction to knowledge graphs and question-answering systems is given. Section 1.2 then introduces the concept of box embeddings and the state-of-the-art Query2box model.

All code is made available here: https://github.com/Jeffrey-Sardina/Query2Geom

### 1.1 Knowledge Graphs and Question-Answering Systems

A Knowledge Graph (KG) is a data structure that represents data objects as nodes, and the relationships between them as labelled directed edges. It is commonly denoted as $G(V, E)$, where $G$ is the graph, $V$ is the set of nodes (or vertices) and $E$ is the set of edges.

The smallest unit of a knowledge graph is a triple $(h, r, t)$, which consists of a head node $h$, a tail node $t$, and a labelled edge $r$ connecting the head to tail. For example, $(Madrid, capitalof, Spain)$ could represent the fact "Madrid is the capital of Spain."

While KGs are very simple in principle, human use is greatly limited by their size: individual KGs can have billions of triples. While this is very noticeable at the scale of the entire "Semantic Web" of interlinked Knowledge Graphs [2], it is also notable among individual KGs [1]. For example, the FB15k dataset has 592,213 triples, its subset FB15k-237 has 272,115 triples [1].

Searching for information in such large KGs is prohibitive. Several approaches seek to address this problem; notably, KG-based Question Answering (QA) systems seek to use machine learning to automatically answer queries posed based on the knowledge in a KG [5]. Such machine learning systems use latent vector representations of the nodes and edges in a knowledge graph to predict the correct answer to a posed query. Several groups have recently investigated this direction, including [3–5].

In this paper, we focus in particular on one particular class of embedding-based QA systems: box embeddings, introduced to the QA task on KGs by Ren et al. [5]. This method of question answering will be treated in detail in the following section.

### 1.2 Box Embeddings and Query2box

Ren et al. was the first group to apply box embeddings to question answering on knowledge graphs [5]. They did this through their model Query2box, which beat state-of-the-art performance when it was published. In short, Query2box embeds nodes in the graph as points in vector space [5]. They then embed questions as boxes whose contents should contain the answers to the question (and only the answers) [5].

This system can not only answer simple questions, but can also handle questions involving logical relationships such as conjunctions by taking the intersection of multiple boxes [5]. For example, it could model the set of nations that border both Spain and France as the intersection of the box representing nations bordering Spain with the box representing nations bordering France.

Query2box makes the choice of calculating the intersection not through geometry, but using an attention mechanism over box embedding vectors [5]. However, they do not compare their system to a geometric-based calculation of box intersections, even though they compare to other, non-geometric baselines [5]. Thus, it remains unclear whether the attention mechanism simply learns to approximate a geometrically precise solution, or whether it is able to produce even better results by attending specifically to features of latent space.

In this paper, we address this gap in our understanding of box embeddings. Our contributions are twofold: first, we propose a new model, Query2Geom, which uses a geometrically exact mechanism to determine the intersection of boxes with fewer trainable parameters, fewer high-level learning components, and a more simple human-interpretation of results. By comparing Query2Geom to Query2box, we find that Query2Geom performs nearly as well, and in some cases better, than Query2box. These results imply that the use of an attention mechanism has few benefits that go beyond simply approximating a geometric solution, while at the same time leading to a higher-parameterised and less-interpretable system.

Second, through an analysis of the difference between attention-based and geometric-based methods, we argue that attention outperforms geometric calculation in some cases because it can attend to latent properties of boxes rather than to their geometry alone. We suggest that the underlying node embeddings used by Query2box do not allow fully expressive box embeddings, and that attention compensates for this inexpressivity in a way geometry cannot. We formalise this description in terms of the concept of "semantic-geometric mismatch". Our results show that semantic-geometric mismatch can simultaneously explain the slight performance loss in Query2Geom compared to Query2box, while also leading directly to the creation of a novel metric of embedding expressivity: the Attention Deviance Metric.

We define the Attention Deviance Metric as the difference between attention-based performance and geometry-based performance, which thus quantifies semantic-geometric mismatch. In this case, embeddings are not fully expressive and are better modelled through attention, which results in greater deviance between the performance of attention-based and geometric-based solutions. When box embeddings are fully expressive this suggests that geometry and semantics of latent space align, removing the deviance between attention and geometric solutions. We call this case semantic-geometric alignment. We then propose future directions for how to more explicitly model for, and take advantage of, this knowledge to further improve box-based QA systems. Finally, we discuss the implications of these findings for the interpretation and formal analysis of attention systems outside of box embeddings alone.

The rest of this paper is organised as follows. Section 2 outlines our methods, notably how we perform geometric box intersection, and how we compare Query2Geom to Query2box. Section 3 presents our results. Section 4 provides a discussion of our findings and our future work. Finally, Sect. 5 concludes the paper.

## 2 Methods

This section is structured as follows: Sect. 2.1 gives an introduction to the box intersection problem. It then describes in detail our method for geometric calculation of the intersection box centre point and offset. Section 2.2 explains out methods for comparing Query2Geom with Query2box.

### 2.1 The Box Intersection Problem

The box intersection problem can be modelled as follows. A box (or hyper-rectangle) exists in $R^n$. It is defined by two vectors: a centre point and an offset, which is a vector made of strictly non-negative values that describes the translation from the centre of the box to one of its vertices. An example of this in $R^2$ is shown in Fig. 1.

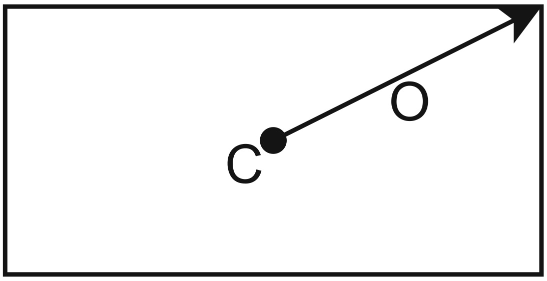

**Fig. 1.** A depiction of a box in $R^2$. $C$ is the box centre, and $O$ is the offset vector.

The Box Intersection Problem is then defined as follows. Start with two boxes, $A$ and $B$, which have centre points $C_A$ and $C_B$ and offsets $O_A$ and $O_B$ respectively. Box intersection attempts to find the so-called intersection box $C$ that is formed by the overlap of boxes $A$ and $B$. An illustration of this in $R^2$ is shown in Fig. 2.

In this paper, we will adopt the following mathematical notations for boxes. For a box $A$, $C_A$ is the vector describing its centre and $O_A$ the vector describing its offset. $C_{A,i}$ is used to describe the $i^{th}$ element (i.e., the $i^{th}$ dimension) of $C_A$; $O_{A,i}$ is defined identically for the offset vector. For examples in $R^2$, $C_{A,x}$ and $C_{A,y}$ will be used for the values in the x and y dimensions of the centre point, and $O_{A,x}$ and $O_{A,y}$ for the x and y dimensions of the offset vector.

It should be noted that since the intersection of any two boxes must be another box, the intersection box is therefore also fully described by a centre point and an offset vector.

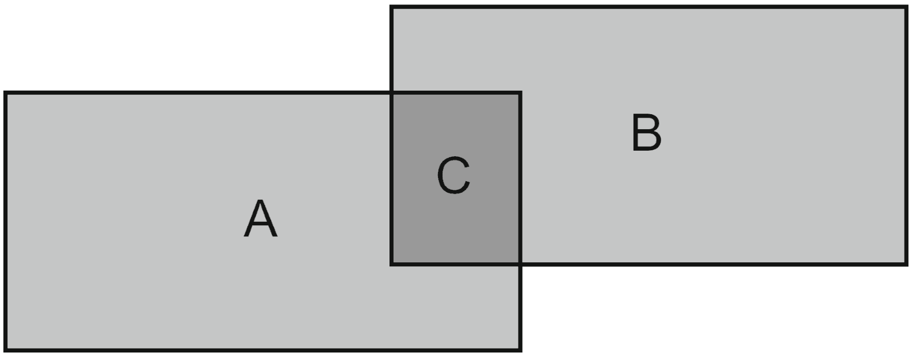

**Fig. 2.** The box intersection problem in $R^2$. Boxes $A$ and $B$ are the boxes being intersected, and $C$ is intersection box formed by their overlap.

**Centre-Point Calculation.** To calculate the intersection of two boxes $A$ and $B$, we start by finding the centre point of the intersection box $C$. To do this, we first note that in each dimension $1..i..n$ of the centre point $C_C$ in $R^n$, that the value of $C_{C,i}$ can be calculated independently of the other values in $C_C$. Thus, we decompose the $n$-dimensional box intersection problem into $n$ 1-dimensional box intersection problems. We call this operation 1-D projection, and it is shown (in the $R^2$ case) in Fig. 3.

We now note that, in the case that $C_{A,i} < C_{B,i}$ the extent of the intersection box in dimension $i$ (i.e., the distance from one hyper-edge to another along the $i$-axis) is bounded by two points: $C_{A,i} + O_{A,i}$ and $C_{B,i} - O_{B,i}$. Note that When $C_{A,i} > C_{B,i}$, these bounds simply reverse to $C_{A,i} - O_{A,i}$ and $C_{B,i} + O_{B,i}$.

Once we have the two end points in dimension i, finding the center point in dimension i is trivial: it is simply the arithmetic mean of the endpoints. Thus, the center point $C_{C,i}$ is given by

$$C_{C,i} = \begin{cases} ((C_{A,i} + O_{A,i}) + (C_{B,i} - O_{B,i}))/2, & \text{if } C_{A,i} < C_{B,i} \\ ((C_{A,i} - O_{A,i}) + (C_{B,i} + O_{B,i}))/2, & \text{otherwise} \end{cases} \tag{1}$$

**Offset Calculation.** Once the centre point of the intersection box, $C_C$ is known, calculation of the offset is trivial. For any value $C_{C,i}$, we know the corresponding lower and upper bound that were used to calculate it as given in the previous section. The offset is simply the positive distance between this centre point and either of the endpoints. To be exact, the offset of the intersection box in dimension $i$, $O_{C,i}$, is given by

$$O_{C,i} = \begin{cases} C_{C,i} - (C_{B,i} - O_{B,i}), & \text{if } C_{A,i} < C_{B,i} \\ (C_{B,i} + O_{B,i}) - C_{C,i}, & \text{otherwise} \end{cases} \tag{2}$$

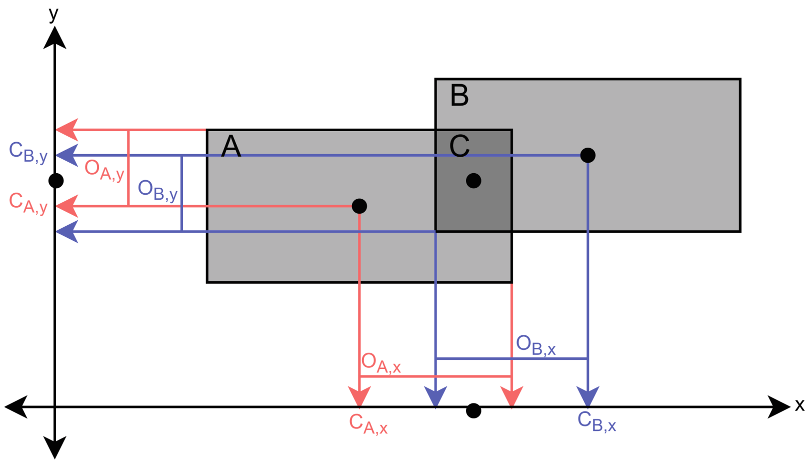

**Fig. 3.** 1-D projections to calculate the box centre in each dimension. $C_{A,x}$ is the x-coordinate of the centre point of box $A$, and $O_{A,x}$ is the x-coordinate of the offset vector of box $A$; $C_{B,x}$ and $O_{B,x}$ are defined likewise for box $B$, and analogous variables are shown in the second dimension along the y-axis.

## 2.2 Comparison Against Query2box

In order to evaluate the performance of our model, we ran both Query2Geom (which uses our geometric box intersections) and Query2box (using the attention-based mechanism for calculating intersection centres and offsets, as in [5]). Both experiments were run with identical hyper-parameters and training configurations, using the setting determined in the original Query2box paper [5]. It is important to note that this includes the fact that both models were trained only on the 1*p*, 2*p*, 3*p*, 2*i*, and 3*i* query types (see the description below), and that remaining query types were only seen in testing. Our experiments were run on the same benchmarking datasets used in the original Query2box paper: FB15k, FB15k-237, and NELL995 [5].

Since Query2box evaluated its performance on a number of different types of queries input to it, we also report performance in all these various types of queries. A brief summary of these query types is given here; for a full explanation, see [5].

Query2box considers 9 types of queries, named 1*p*, 2*p*, 3*p*, 2*i*, 3*i*, *ip*, *pi*, 2*u*, and *up*. In this notation, *p* represents the "projection" operator, which is how Query2box models traversing relationship embeddings of the KG. *i* represents the intersection operator, which is how the logical conjunction is modelled. *u* represents the union operator, which is how logical disjunction is modelled. Numbers indicate the number of each operator applied (i.e. 2*u* means two union operations), and the order of elements describes the order in which the operators are used (i.e., *pi* means that a projection is followed by an intersection).

In the remainder of this paper, this notation will be used when discussing query types.

## 3 Results

This section is structured as follows: Sect. 3.1 compares the overall model complexity of Query2Geom and Query2box. Section 3.2 gives an analysis of the comparative performance of both models. Section 3.3 introduces our Attention Deviation metric and the concepts of semantic-geometric alignment and mismatch.

### 3.1 Model Complexity

The sizes of both models in terms of the number of parameters are summarised in Table 1. The relative decrease in the number of parameters used in Query2Geom compared to Query2box is also given.

**Table 1.** A summary of the number of model parameters when trained with an embedding dimension of 400.

| Dataset | #params (Query2box) | #params (Query2Geom) | % reduction |
|---|---|---|---|
| FB15k | 8772400 | 8132400 | 7.30% |
| FB15k-237 | 6821200 | 6181200 | 9.38% |
| NELL995 | 26304400 | 25664400 | 2.43% |

It is critical to note that the simplifications in Query2Geom are not simply reductions in parameter usage. The parameter reduction comes from entirely replacing two learning components of the model – namely intersection box centre calculation and offset calculation – with fixed geometric formulas. It is a reduction in the number of higher-level learning components of the original model, which translates to simplification at the conceptual, interpretability, and architectural levels.

### 3.2 Model Performance

Performance was measured by four different scores: MRR (mean reciprocal rank), Hits@1, Hits@3, and Hits@10. All of these scores are calculated based on the ranking of correct responses among incorrect ones. MRR is the mean reciprocal rank of the correct answer among incorrect answers; the three Hits@k measures are the proportion of correct answers in the top k elements of the ranking.

The scores of both models by each of these metrics are summarised in Table 2, Table 3, and Table 4.

In general, Query2Geom matches or nearly matches the performance of Query2box on all datasets and query types examined. The two exceptions to this

**Table 2.** The performance of Query2Geom vs Query2box on each query type for FB15k. Performance is measured in terms of MRR (mean reciprocal rank), Hits@1, Hits@3, and Hits@10.

| FB15k | | | | | |
|---|---|---|---|---|---|
| Query type | Model | MRR score | Hits@1 score | Hits@3 score | Hits@10 score |
| 1p | Query2Geom | 0.650267 | 0.489999 | 0.780970 | 0.903410 |
| | Query2box | 0.660733 | 0.504797 | 0.787674 | 0.905719 |
| 2p | Query2Geom | 0.374285 | 0.275038 | 0.411344 | 0.572442 |
| | Query2box | 0.378315 | 0.278593 | 0.415397 | 0.579878 |
| 3p | Query2Geom | 0.273310 | 0.182618 | 0.302931 | 0.447608 |
| | Query2box | 0.277408 | 0.186537 | 0.307702 | 0.452624 |
| 2i | Query2Geom | 0.466711 | 0.324910 | 0.555033 | 0.720269 |
| | Query2box | 0.492412 | 0.343777 | 0.590193 | 0.756878 |
| 3i | Query2Geom | 0.559498 | 0.417586 | 0.658477 | 0.809164 |
| | Query2box | 0.604431 | 0.459966 | 0.710518 | 0.852715 |
| ip | Query2Geom | 0.183677 | 0.111511 | 0.199075 | 0.322121 |
| | Query2box | 0.192262 | 0.117338 | 0.211144 | 0.337771 |
| pi | Query2Geom | 0.335959 | 0.217770 | 0.394403 | 0.558723 |
| | Query2box | 0.349359 | 0.226694 | 0.410136 | 0.583978 |
| 2u | Query2Geom | 0.460892 | 0.269328 | 0.600964 | 0.805769 |
| | Query2box | 0.476807 | 0.289031 | 0.613005 | 0.811203 |
| up | Query2Geom | 0.298903 | 0.202786 | 0.326896 | 0.498826 |
| | Query2box | 0.303698 | 0.209297 | 0.328150 | 0.502811 |

**Table 3.** The performance of Query2Geom vs Query2box on each query type for FB15k-237. Performance is measured in terms of MRR (mean reciprocal rank), Hits@1, Hits@3, and Hits@10.

| FB15k-237 | | | | | |
|---|---|---|---|---|---|
| Query type | Model | MRR score | Hits@1 score | Hits@3 score | Hits@10 score |
| 1p | Query2Geom | 0.402108 | 0.278309 | 0.468142 | 0.634959 |
| | Query2box | 0.403250 | 0.279491 | 0.468547 | 0.638436 |
| 2p | Query2Geom | 0.225566 | 0.144491 | 0.246415 | 0.386744 |
| | Query2box | 0.228106 | 0.147005 | 0.249112 | 0.392738 |
| 3p | Query2Geom | 0.175492 | 0.108264 | 0.189519 | 0.312062 |
| | Query2box | 0.176689 | 0.108015 | 0.190522 | 0.314173 |
| 2i | Query2Geom | 0.237886 | 0.124677 | 0.283483 | 0.465320 |
| | Query2box | 0.274497 | 0.157053 | 0.324284 | 0.513665 |
| 3i | Query2Geom | 0.338280 | 0.216228 | 0.401429 | 0.571145 |
| | Query2box | 0.378354 | 0.248346 | 0.452466 | 0.621043 |
| ip | Query2Geom | 0.099940 | 0.052975 | 0.100593 | 0.188812 |
| | Query2box | 0.106850 | 0.056463 | 0.111404 | 0.201593 |
| pi | Query2Geom | 0.171607 | 0.095063 | 0.193016 | 0.315974 |
| | Query2box | 0.182645 | 0.100906 | 0.205410 | 0.340416 |
| 2u | Query2Geom | 0.201138 | 0.087873 | 0.242313 | 0.432366 |
| | Query2box | 0.206530 | 0.094251 | 0.247073 | 0.434787 |
| up | Query2Geom | 0.176320 | 0.103061 | 0.188334 | 0.329051 |
| | Query2box | 0.180102 | 0.105682 | 0.191140 | 0.328722 |

**Table 4.** The performance of Query2Geom vs Query2box on each query type for NELL995. Performance is measured in terms of MRR (mean reciprocal rank), Hits@1, Hits@3, and Hits@10.

| NELL995 | | | | | |
|---|---|---|---|---|---|
| Query type | Model | MRR score | Hits@1 score | Hits@3 score | Hits@10 score |
| 1p | Query2Geom | 0.411444 | 0.227585 | 0.550640 | 0.710378 |
| | Query2box | 0.414124 | 0.229345 | 0.556521 | 0.711783 |
| 2p | Query2Geom | 0.225108 | 0.128063 | 0.263442 | 0.418847 |
| | Query2box | 0.228444 | 0.131157 | 0.266658 | 0.423136 |
| 3p | Query2Geom | 0.204901 | 0.123397 | 0.230799 | 0.360781 |
| | Query2box | 0.208871 | 0.128412 | 0.234453 | 0.362386 |
| 2i | Query2Geom | 0.271019 | 0.152069 | 0.319716 | 0.524850 |
| | Query2box | 0.289603 | 0.162182 | 0.348049 | 0.553331 |
| 3i | Query2Geom | 0.398097 | 0.275297 | 0.453691 | 0.643913 |
| | Query2box | 0.421173 | 0.290373 | 0.488008 | 0.678520 |
| ip | Query2Geom | 0.117654 | 0.066895 | 0.123042 | 0.215243 |
| | Query2box | 0.124954 | 0.072257 | 0.131605 | 0.226891 |
| pi | Query2Geom | 0.200642 | 0.120546 | 0.225809 | 0.354733 |
| | Query2box | 0.192567 | 0.116442 | 0.213335 | 0.339977 |
| 2u | Query2Geom | 0.261958 | 0.099772 | 0.367293 | 0.576311 |
| | Query2box | 0.264654 | 0.100538 | 0.371798 | 0.579765 |
| up | Query2Geom | 0.154615 | 0.078754 | 0.163667 | 0.322732 |
| | Query2box | 0.156213 | 0.079159 | 0.167033 | 0.329692 |

are the $2i$ and $3i$ query types, in which case Query2box outperforms Query2Geom by a wider margin relative to other query types on each dataset. On the other query types using intersections, *ip* and *pi*, Query2Geom has a much smaller performance gap. That gap is almost always present, with the one major exception being Query2Geom out-performing Query2box on *pi* queries on NELL995. Overall, this suggests that the attention mechanism employed in Query2box is able to find slightly more performant intersections than the exact geometric values by attending to the properties of latent space.

On the other hand, Query2Geom uses between 2.5% and 10% fewer parameters than Query2box on the datasets tested, and in most cases performs almost identically to Query2box; notably, see $1p$, $2p$, $3p$, $2u$, and *up* queries on all datasets.

## 3.3 The Attention Deviation Metric and Semantic-Geometric Alignment and Mismatch

Our final result is the derivation of two critical ideas: the Attention Deviation Metric (ADM) and the concept of Semantic-Geometric Alignment and Mismatch.

In cases where there was a discrepancy between the performance of Query2Geom (which attempts to enforce semantic-geometric alignment) and Query2box (which allows semantic-geometric mismatch), we note that Query2box generally performed slightly better. This is clear evidence for the presence of semantic-geometric mismatch in Query2box: if its attention mechanism only served to approximate a geometric solution, then it would have been perfectly matched or outperformed by Query2Geom's exact geometric solution, as that exact solution would have less error than the solution found by the attention mechanism. The deviation between these two scores can then be explained by the attention mechanism performing a "latent intersection" rather than a geometric one, using the hidden properties of boxes to yield more accurate results.

This leads us to the concepts of semantic-geometric alignment and mismatch. When box intersections are geometrically exact, then a box represents all answers to a query, and an intersection of two boxes represents all answers that satisfy both queries they represent. At a mathematical level, this means that the model attempts to enforce perfect alignment between the latent properties of the box embeddings and the geometric properties of embedding space. In the case that the embedding system is able to fully capture the semantics of the knowledge graph, it should produce such an alignment. Since the latent properties of the embeddings are representations of the semantics of the Knowledge Graph, we call this case "semantic-geometric alignment". In other words, when there is semantic-geometric alignment, the semantics of the knowledge graph relevant to the question-answering task are contained within the geometry of their embeddings.

When attention is used rather than geometry to determine box intersection, semantic-geometric alignment is not enforced. Instead, the attention mechanism is encouraged to examine the latent features of box embeddings, and to give higher or lower weights to various elements of latent space that it finds correlate better or worse to the model's training performance – even when that leads to geometrically inexact intersections. We call these intersections latent intersections to distinguish them from exact geometric intersections. The use of latent intersections leads to semantic-geometric mismatch. We note that the primary drive for using latent intersections would be in the case that the embeddings are not fully expressive, since semantic-geometric alignment would not be able to hold for such a system.

Looking back at the gap between Query2box and Query2Geom, we formally describe this deviation as one caused by semantic-geometric mismatch that drives Query2box's attention mechanism to use slightly higher-performing latent intersections rather than approximating geometric ones. This leads us directly to the Attention Deviation Metric. ADM seeks to measure to what extent the latent intersections are able to learn more than geometric intersections: in other words, to quantify the extent of semantic-geometric mismatch. The Attention Deviation Metric is given by the following formula:

$$ADM = score(Attn) - score(Exct)$$

where *score* is a score function such as MRR, *Attn* is a model using attention-based calculation (such as Query2box) and *Exct* is a model replacing attention with an exact mathematical calculation (such as Query2Geom).

Here, using MRR as our score function, Query2box as *Attn* and Query2Geom as *Exct*, we can calculate the ADM between these two values. For example, looking at $2i$ queries on FB15k-237, we can calculate

$$ADM = MRR(Query2box) - MRR(Query2Geom)$$

$$... = 0.274497 - 0.237886 = 0.036611$$

Similarly, for $1p$ queries on FB15k-237, the associated ADM value is 0.001142. This means that for FB15k-237, the use of latent intersections rather than geometric ones has a far ( 30x) higher impact on $3i$ queries than on $1p$ ones. This pattern generalises our previous observation that Query2box tends to perform better on queries with intersections compared to Query2Geom.

## 4 Discussion and Future Directions

We succeed in generally matching the performance of Query2box with a much simpler model, Query2Geom, that uses exact geometric calculations instead of an attention mechanism to calculate box intersections. Query2Geom has several benefits – many fewer trainable parameters and fewer high-level learning components. This results in a lighter demand on resources and a better ability to scale. However, it also means that the model is easier to interpret, since it is based on semantic-geometric alignment unlike Query2box, which is based on semantic-geometric mismatch.

The necessity of using a latent intersection fundamentally implies that the boxes constructed by Query2box are not fully expressive; i.e. that they do not fully capture the concept grouping they were designed to model. After all, if the boxes were fully expressive, then the power of the geometric intersection operator would approach that of the attention intersection operator, because perfect semantic-geometric alignment would hold in latent space. This effect is quantified by our Attention Deviation Metric, which captures the limits on expressivity of box embeddings through using semantic-geometric mismatch.

We propose that Query2Geom cannot reach full semantic-geometric alignment because the underlying node embeddings are not able to capture the full semantics of the Knowledge Graph. More expressive embeddings would result in semantic-geometric alignment, which would eliminate the small remaining benefit of attention. Creating such an aligned system and determining its properties is left as a future direction.

Beyond the realm of box embeddings, our work has a critical point to make about the function of attention in general. The clear presence semantic-geometric mismatch when attention is used implies that attention does not serve to simply approximate exact geometric (or other mathematical) functions. Instead, attention exists to learn how to use latent entity representations that are not captured by geometry and exact formulas. We hypothesize that the ADM presented here will generalise to other attention mechanisms, and that the extent of semantic-geometric alignment or mismatch in a model can be calculated by ADM with attention-free alternatives. Exploring this hypothesis is left as a future direction.

## 5 Conclusion

In this paper, we introduced Query2Geom, a modification of Query2box that replaces its attention-based box intersection system with an exact geometric one. Our results indicate that both models perform very similarly, but that Query2box slightly outperforms Query2Geom because its attention mechanism allows it to attend to aspects of latent space that are not captured in a pure geometric model – a case we formalise as semantic-geometric mismatch. This led us to propose the Attention Deviation Metric, which models the expressiveness of a box embedding model by the performance lost when replacing attention-based intersection with precise geometric calculations of box intersections.

We leave as a future direction applying the Attention Deviation Metric to estimate the performance of other box embedding models, and other attention-based models more generally. Finally, we propose that research in this direction will not only lead to improvements in model performance, but also lead to increases in training resource- and time- efficiency.

**Acknowledgements.** This research was conducted with the financial support of Science Foundation Ireland D-REAL CRT under Grant Agreement No. 18/CRT6225 at the ADAPT SFI Research Centre at Trinity College Dublin, together with sponsorship of Sonas Innovation Ireland. The ADAPT SFI Centre for Digital Content Technology is funded by Science Foundation Ireland through the SFI Research Centres Programme and is co-funded under the European Regional Development Fund (ERDF) through Grant # 13/RC/2106_P2.

## References

1. Ali, M., et al.: Bringing light into the dark: a large-scale evaluation of knowledge graph embedding models under a unified framework. IEEE Trans. Pattern Anal. Mach. Intell. **44**(12), 8825–8845 (2021). https://doi.org/10.1109/TPAMI.2021.3124805
2. Bizer, C., Heath, T., Berners-Lee, T.: Linked data: the story so far. Int. J. Semant. Web Inf. Syst. **5**, 1–22 (2009). https://doi.org/10.4018/jswis.2009081901
3. Gu, K., Miller, J., Liang, P.: Traversing knowledge graphs in vector space (2015). https://doi.org/10.18653/v1/D15-1038
4. Hamilton, W.L., Bajaj, P., Zitnik, M., Jurafsky, D., Leskovec, J.: Embedding logical queries on knowledge graphs. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS 2018, Curran Associates Inc., Red Hook, NY, USA, pp. 2030–2041 (2018)
5. Ren, H., Hu, W., Leskovec, J.: Query2box: reasoning over knowledge graphs in vector space using box embeddings. In: International Conference on Learning Representations (2020). https://openreview.net/forum?id=BJgr4kSFDS