We really appreciate the reviewers' time and effort. Thank you for your detailed feedback, thoughtful suggestions and valuable recommendations for improving the paper! The reviewers appreciate our strong empirical results and mainly asked us to better situate our work. We respond in more detail below, and will take all comments into account in our revised version. Additionally, we will shortly release open-source PyTorch implementations of the Hyperbolic Graph Convolution Networks (HGCN) model and baselines, along with our detailed reproducible training setup.

Contributions (R1, R2, R3): We sincerely thank R1 for their careful read and for pointing out ambiguities in our paper. First, our paper is an empirical paper. Our goal is to develop practical techniques that can improve the predictive performance of recently-developed Graph Convolutional Networks (GCNs) using ideas from hyperbolic geometry. Our main result is that HGCN achieves error reduction of up to 63.1% in ROC AUC for link prediction and of up to 47.5% in F1 score for node classification. Moreover, using standard notions from hyperbolic geometry (Gromov's δ-hyperbolicity), we show that performance is indeed improved when the underlying graph is more "hyperbolic-like".

Simply setting GCN variables to be optimized in hyperbolic space does not yield good performance. Our work validates that three algorithmic ideas based on hyperbolic geometry are important to obtaining predictive accuracy and good runtime performance: trainable curvature, attention-based hyperbolic aggregation, and optimization on the Lorentz (hyperboloid) manifold. Our aggregation method relies on two crucial techniques which result in improved performance compared to standard aggregation in the tangent space at the origin: (1) aggregation is performed at the *local* tangent space of each point, which better approximates the local hyperbolic geometry; (2) attention scores are computed from the origin, which allows HGCN to capture node hierarchies. Furthermore, trainable curvature intuitively helps find the right amount of curvature, and potentially alleviates numerical errors that might arise from limited machine precision. We show in ablation analysis that these algorithmic contributions result in up to 9.9% absolute gain compared to simple GCN in hyperbolic space, an improvement that is larger than what any Euclidean GCN variant achieves.

Presentation (R1, R2, R3): The results in Section 3 are not considered part of our contribution and we now realize that we could make our claims and presentation more crisp. In our updated draft, we will state known facts from hyperbolic geometry as propositions rather than corollaries, move standard results to the Appendix as suggested by R2, and more carefully cite the related literature, including [1], as suggested by R1. We have also fixed the typo in Equation 15 in the Appendix pointed out by R1 and we confirm that we have been using the correct formulation of parallel transport in our experiments. Indeed, we had run unit tests verifying that points are mapped to the correct tangent space. We also thank R1 for noticing the notation error regarding the hyperbolic radius $i\sqrt{K}$, and we will make this consistent in the revised version.

We thank R2 for the thorough feedback and will clarify our intuitive explanation on the hyperbolic volume growth property in line 27, which was meant to illustrate the fact that the volume of balls in Euclidean space grows polynomially with respect to the radius, while in hyperbolic space it grows exponentially. We will also replace the hyperboloid manifold by the Lorentz manifold in the revised version.

R3 also asks about the correctness of Equation 10, which maps tangent spaces with different curvatures. In order to apply the exponential map at the north pole, we need to make sure that points are located in the corresponding tangent space. Fortunately, tangent spaces of the north pole are shared across hyperboloid manifolds that have different curvatures, making equation 10 mathematically correct as long as $\sigma(0) = 0$, which is true for the ReLU activation. We will make this more explicit in the revised version.

Experiments (R1, R2): R1 rightly points out that the number of hyper-parameters should be the same for fair comparison between models. In our experiments, we were very careful about this and we ensured fairness of all methods by controlling the number of hyper-parameters and trainable parameters. We will clarify this detail in the revision. In particular, curvatures in HGCN are trainable parameters which are therefore not subject to hyper-parameter search. Regarding bias and temperature hyper-parameters, we consistently used the Fermi-Dirac decoder to predict link probabilities for our method and all baselines (hyperbolic and Euclidean), observing that it performs the same as the standard dot product decoder commonly used in Euclidean baselines.

R2 requests clarification regarding the stability of the hyperboloid model compared to the Poincaré model. In our experiments, we used both models observing that they achieve similar performance, but that the hyperboloid model offers more stable optimization. This statement is also supported by the fact that the Poincaré distance function is numerically unstable due to the denominator term, confirming a previous observation in [2]. We will clarify this point in the revision.

Connection to prior work (R2): We appreciate R2's suggestion to discuss the connections to Hyperbolic Neural Networks (HNN). The main differences are the use of trainable curvatures, the attention-based hyperbolic aggregation mechanism, and the use of hyperboloid model. In the revised manuscript, we will further highlight and stress these key components and add an experiment using HNN in the hyperboloid model, as suggested by R2.

[1] M. Law, R. Liao, J. Snell, and R. Zemel. "Lorentzian Distance Learning for Hyperbolic Representations." ICML 2019.
[2] M. Nickel, and D. Kiela. "Learning Continuous Hierarchies in the Lorentz Model of Hyperbolic Geometry." ICML 2018.