We thank the reviewers for useful and detailed feedback, which has helped us to improve our paper. We will release our code soon. First, we will respond to questions common among the reviewers, and then address individual concerns.

All reviewers asked about the higher-dimensional case. To clarify, any tiling with all-congruent tiles (even in higher dimensions) can be represented as a subset of the set of isometries. This subset being a group, as for L-tilings is desirable because the symmetries involved simplify the analysis and representation of the tiles. For H-tilings, the subset is not a group, and this can't be avoided in higher dimensions because of Coxeter's result [6]. Nevertheless H-tilings can still be used for learning, as we show in Section 6. Despite their inapplicability to high-dimensional spaces, L-tilings are still desirable in cases where we do want to learn/embed over 2D space or the Cartisian product of 2D spaces, as was done in Gu et al [14]. We will update our manuscript to clarify this point.

R2 and R3 correctly pointed out that currently our methods apply to loss functions that depend on distances. However, most embedding tasks such as word embedding and link prediction in networks, do depend on distances and can be computed efficiently with our methods. Extension to other loss functions will be future work.

Q1 from R1: Regarding line 114, "its curvature still . . . "? —–A1: We meant to say that the Riemannian metric of the model becomes large and poorly conditioned, and not the manifold curvature. We have fixed this.

Q2 from R2: In line 86, "This suggests that . . . " doesn't come as a logical consequence. —–A2: We misplaced this sentence; it was intended to refer to the comparison between Poincare and Lorenz models (two different numerical models) in [19], not to Gu et al [14].

Q3 from R2: Does the numerical instability affect the performance of models for different tasks mentioned in line 91? —–A3: We believe numerical stability is an issue for different tasks based on communications with authors of some cited papers. We plan to cut this claim and leave that evaluation to future work.

Q4 from R2: In line 162, "Importantly, any element ...", is this a fact for all Fuchsian matrices? —–A4: No, it's not generally true for Fuchsian groups, we choose the generators (integer matrix) in Definition 1 to make it happen.

Q5 from R3: Clarify the meaning of SGD and RSGD? —–A5: In this paper, SGD uses the Euclidean gradient within the model, while RSGD transforms the Euclidean gradient to a Riemannian gradient, and uses the exponential map on the manifold to update parameters. All models including baselines are trained with RSGD except that we specifically train a L-tiling model with SGD for comparison as mentioned in Line 282.

Q6 from R3: Explain algorithms 1,2 and the minimization of W in algorithm 2, how computationally expensive it is (explain theoretically and empirically)? What are the training time? —–A6: As Theorem 3 and 6 states, algorithm 1 maps a point in the Lorentz model to a point (U, u) in the L-tiling model, where u is unique in F, so algorithm 1 outputs the solution U of the minimization problem in algorithm 2. In the proof of algorithm 2, $W = U^{-1}V$ is a middle variable for convenience and different from the W in the minimization, we will change the symbols and rewrite this part. The computational complexity of algorithm 1 is linear in the distance of the point from the origin as shown in Theorem 3. Empirically, for an existing embedding of Gr-QC dataset (4158 nodes), in which the largest and average absolute value are 2.05e+10 and 1.48e+07, it will take 0.92 seconds to solve all minimization problems. But for training, points are initialized near the Origin, the minimization problem is solved once the point is out of F, so typically it will finished within 3 steps. As for the training time, take the learning of 2D embeddings for Wordnet Verbs for example, same as released baselines' code, which trains the embedding on CPU, we trained 5 models for 1000 epochs, here is the time: L-tiling-SGD: 27079s, L-tiling-RSGD: 18028s, Lorentz: 7867s, Poincare: 20422s, H-tiling; 18388s.

Q7 from R3: How to translate from a given matrix U to the VBW encoding? —–A7: If U is given, consider the point (U, O) in the L-tiling model, which is $x = LUL^{-1}O$ in the Lorentz model, then we can map x to (U', u') with algorithm 1, where we choose a generator at each step, then we can store a generator order string from algorithm 1, with which we can reconstruct U'. Since each point in the Lorentz model will be mapped to a unique point in F, also x can be mapped to (U, O) and (U', u'), so $u' = O$. The question is whether $U = U'$, consider $LUL^{-1}O = LU'L^{-1}O$, which leads to $(LU'^{-1}UL^{-1} - I)O = 0$, as Line 37-39 in appendix shows, we prove that $LU'^{-1}UL^{-1} = I$, then $U = U'$. Hence, given U, we can get its generator order string, then we can get the VBW encoding accordingly.

Q8 from R3: Can you prove the statement in line 252: each square is isometric to every other square? —–A8: We've called them squares even though in the hyperbolic metric they bear no resemblance to squares. See page 95-98 of Cannon et al [4] for an introduction of these isometries and "squares" with a nice graph; We will more clearly reference this in our updated manuscript.

We will improve write-up and methods in the following way: add mathematical concepts like Fuchsian group into appendix, add a section about learning in appendix to explain more extensively of section 5 and 6. For experiments, add product of baseline models for dimensions 4 and 6, add confidence intervals to the results, add a training detail section in appendix. Also, fix some typos, statements and inconsistencies in the bibliography.