We thank the reviewers for taking the time to conduct a second review of our manuscript. We have compiled comments from both reviewers below (in bold) together with our response and actions taken (in italics).

I appreciate the extended explanations on this modelling choice, but according to the authors' reply to the reviewers, I am still not sure that we are on the same page in terms of our previous questions. The authors justify their choice saying that if they wanted to use a regression network, they would have to apply a CNN like the one in Jouvet et al. (2021). I don't understand this comparison between different architectures, since any neural network architecture can be used for both classification and regression. Even U-Net can be used for regression, it would be as simple as changing the very last layer, with a single neuron and a linear activation function, and replacing the loss function for something like RMSE or MAE. What defines a network as a classification or regression one is not its architecture, but its output layer and loss function. It is unclear if the authors have attempted this extremely simple change, or if they have tried other different architectures. Directly adjusting U-Net for a regression problem would the most straightforward and fair way to test this. I understand that if this current architecture works, it makes sense to keep it. But I'm also worried that such an architecture might end up being used as a reference for this sort of tasks, which is clearly and overcomplication of the problem. If the U-Net and the autoencoder where optimized at the same time (i.e. with a single loss at the output of the autoencoder), this would already make more sense, since backpropagation would take place as a regression problem. However, destroying information to train a first classification network and then asking another network to undo the damage looks problematic to me. In order to avoid a sterile debate over this, I ask the authors the following:

• Can you please clearly explain what exact architecture and changes you applied to the U-Net in order to use it as a regression problem? This should be clearly explained in the Appendix. I would strongly encourage the authors to test the very simple changes proposed above to use the U-Net for regression. Simply change the output layer and the loss function. This should be less than 5 lines of code.

• You should clearly state in the methods section and the abstract that this is a regression problem. You can explain that this more complex combined method of classification + regression works well here, but the reader should know that this is an exception, not the norm.

We accept that in our previous review we did not unequivocally address the reviewers concerns with regards to our choice of network architecture and we trust that the additional details provided in the manuscript and extended testing can resolve this. We tested many architectures during the development of MELTNET and in the manuscript we have added a direct comparison with two alternatives, including the exact approach suggested by the reviewer in their most recent comments. As we now show in detail in a new appendix and figure, MELTNET clearly outperforms both a CNN-type regression architecture and a regression version of the UNET-type architecture that we use within MELTNET.

We have a different perspective to the reviewer on the task that MELTNET is solving. The reviewer continues to refer to this as a regression problem and instead we view our method as performing a segmentation task (the heavy lifting of prediction, done using a variant of RESUNET), followed by denoising (where the low resolution blocky nature of the melt rate labels is converted to highly resolved melt rates, done using a denoising autoencoder). In this way, the approach is very different from treating the problem directly as a regression task, with both advantages and disadvantages. In this case, it seems that the advantages outweigh the disadvantages, and the performance is

consistently better than directly solving a regression problem. Thus, the approach is not a roundabout and long-winded regression, but something different. If it were simply an overcomplicated way of doing regression then we agree with the reviewer, information is lost in the process and so surely there is a better way, and yet here our method performs better - because it is doing something entirely different.

Ultimately, we do not believe the choice of deep learning architecture should be the focus of this paper and our aim is not to show that splitting this melt rate prediction task into two networks is a silver bullet for other similar problems. For this reason, we have changed the title of our paper to avoid any confusion that may have arisen on the paper's goal. That being said, we do not understand the reviewer's insistence that this method is a clear overcomplication and their apparent concern that our approach might be tried in other contexts. The statement that a combination of classification and regression works well here being an exception rather than the norm is curious since to our knowledge the reviewer has not tested this approach themselves. Both before and since receiving reviews on our initial manuscript submission we have extensively tested alternatives and cannot find anything that outperforms the methodology that we present in the paper. Therefore, while we do make it clear that this is a novel method that has not been used before, we see no justification to specifically discourage the reader from attempting to use a similar approach for their own work. The field of deep learning is evolving rapidly and no doubt an architecture exists or will be developed soon that outperforms the method presented in the paper, but that has no bearing on our results or conclusions.

One aspect that I admit escaped my first review is the absence of a test dataset. Currently the authors are splitting their total dataset into 90% training and 10% validation. Then, they tune the hyperparameters of the networks based on the results of the validation dataset, but no independent assessment of model performance is made. This is highly problematic, since the model hyperparameters are overfitting the validation dataset, which is also used to assess the final model performance. The usual and recommended practice in machine learning is to split the dataset into train, validation and test datasets. The reasons to use an independent test dataset are widely explained in the literature, so I leave a reference here for more details (see points 2.2 and 3.1): Lones (2022), How to avoid machine learning pitfalls: a guide for academic researchers. Before publication, the authors should repeat the training process, putting aside a test dataset. A possible division could be 70% training, 20% validation and 10% test. The authors are free to choose these ratios, but I would encourage them to use at least 10% on test. The split between these 3 datasets should be done before training, and the test dataset should be kept aside and not used until the very last moment, which will serve to have the "real" model performance. The final performance on the test dataset should be at least slightly lower than the one on the validation dataset (if the model is generalizing correctly). All figures displaying model performance should be updated with the results of the test dataset. In order to allow a fair comparison against PICO and PLUME, these dataset divisions should also be applied throughout all models (particularly for Fig 4).

To address this criticism we have conducted a further ~1000 NEMO simulations, comprising approximately 10% of our total data, that is used as our 'test' set as suggested by the reviewer. These simulations are not seen during training, nor were they used to optimise model hyperparameters. We now use this test dataset, rather than the previously defined validation set, to evaluate the performance of MELTNET, together with the PICO and PLUME parameterisations. We have updated all relevant figures based on this new test set. We have also clarified this split in our

dataset wherever relevant in the text. Our finding shows that this change has no discernible impact on the model performance and does not affect any of the conclusions of our paper.

- Line 92: Please cite the GGL90 scheme - I'm assuming this is what is being used (Gaspar et al, 1990, https://doi.org/10.1029/JC095iC09p16179)

The turbulent kinetic energy scheme is indeed based on Gaspar et al, (1990) but has since been heavily modified, see p54 of Madec et al. (1998) or 9.1.3 in Madec et al. (2019) for details. We have added a reference to Madec et al. (1998).

- Line 95: Please cite the Redi scheme, again I'm only assuming this is what's being used. (Redi, 1982, http://journals.ametsoc.org/view/journals/phoc/12/10/1520-0485_1982_012_1154_oimbcr_2_0_co_2.xml) If Gent and McWilliams, (1990, http://journals.ametsoc.org/view/journals/phoc/20/1/1520-0485_1990_020_0150_imiocm_2_0_co_2.xml) is being used please cite that too.

L95 was discussing the three-equation parameterization but we think the reviewer here was asking about subgrid scale eddy parameterisations such as GM. In this instance we did not use a GM-like parameterisation. We do not think this is unusual at these latitudes with eddy permitting resolution.

- Line 101: Please either say that the vertical spacing is 45.5m or that the vertical levels are evenly spaced.

We have specified that the vertical levels are evenly spaced

- Appendix C and Figure C1: I think the manuscript is greatly improved by adding the 2 appendices with details on the neural network architecture - this makes the results more reproducible without being a distraction in the main text. Two very minor suggestions here:
1) how big are the convolutional layers used in the Auto Encoder (for inverse classification)? This could be added to the figure or to the text in this appendix.
2) the regularization parameter amplitude for this architecture could be added in line 574 for completeness.

We have added this information to the manuscript as suggested by the reviewer