WHAT IS CLAIMED IS:

1. A system for reducing dataset biases in natural language inference tasks using unadversarial training, the system comprising:

at least one processor programmed or configured to:

perform an unadversarial training procedure to train a machine learning model to provide a trained machine learning model, wherein, when performing the unadversarial training procedure, the at least one processor is programmed or configured to:

receive a training dataset comprising a plurality of training samples, each training sample of the plurality of training samples includes transaction data associated with an electronic payment transaction;

generate a noise vector for the plurality of training samples based on a first uniform distribution;

perturb each training sample of the plurality of training samples based on the noise vector to provide a plurality of perturbed training samples;

obtain a gradient between a first output of the machine learning model that results from inputting each training sample of the plurality of training samples and a second output of the machine learning model that results from inputting each perturbed training sample of the plurality of perturbed training samples;

generate an updated noise vector based on the gradient;

perturb each training sample of the plurality of training samples based on the updated noise vector to provide a second plurality of perturbed training samples; and

update a model weight of the machine learning model based on the second plurality of perturbed training samples to provide the trained machine learning model;

wherein, the trained machine learning model is a first trained machine learning model that exploits bias features of the training dataset to make predictions, and wherein the at least one processor is further programmed or

configured to:

train a second machine learning model using a Product of Experts (POE) procedure based on the first trained machine learning model to provide improved predictions as compared to the first trained machine learning model.

2. The system of claim 1, wherein, when generating the noise vector for the plurality of training samples, the at least one processor is programmed or configured to:

initialize the noise vector with a uniform distribution, wherein, when initializing the noise vector, the at least one processor is programmed or configured to:

multiply the first uniform distribution by a predefined radius to provide a second uniform distribution; and

divide the second uniform distribution by a square root of an input embedding size of the machine learning model to provide an initial value of the noise vector.

3. The system of claim 1, wherein, when generating the updated noise vector, the at least one processor is programmed or configured to:

multiply the gradient by a step size.

4. The system of claim 1, wherein, the at least one processor is further programmed or configured to:

restrict a value of the updated noise vector based on a predefined radius to provide a second updated noise vector; and

wherein, when perturbing each training sample of the plurality of training samples based on the updated noise vector to provide the second plurality of perturbed training samples, the at least one processor is programmed or configured to:

perturb each training sample of the plurality of training samples based on the second updated noise vector to provide the second plurality of perturbed training samples.

5. The system of claim 1, wherein, when training the second machine learning model using the POE procedure, the at least one processor is further programmed or configured to:

generate an unnormalized output from the first trained machine learning model based on the plurality of training samples of the training dataset;

generate an unnormalized output from the second machine learning model based on the plurality of training samples of the training dataset;

combine the unnormalized output from the first trained machine learning model and the unnormalized output from the second machine learning model to provide a combined unnormalized output; and

update a model weight of the second machine learning model based on the combined unnormalized output to provide the second trained machine learning model.

6. The system of claim 5, wherein, when generating the unnormalized output from the first trained machine learning model based on the plurality of training samples of the training dataset, the at least one processor is programmed or configured to:

generate an output of the first trained machine learning model using a first logits function based on the plurality of training samples of the training dataset;

wherein, when generating the unnormalized output from the second machine learning model based on the plurality of training samples of the training dataset, the at least one processor is programmed or configured to:

generate an output of the second trained machine learning model using a second logits function based on the plurality of training samples of the training dataset;

wherein, when combining the unnormalized output from the first trained machine learning model and the unnormalized output from the second machine learning model to provide the combined unnormalized output, the at least one processor is programmed or configured to:

combine the output of the first trained machine learning model using the first logits function based on the plurality of training samples of the training

dataset and the output of the second machine learning model using the second logits function based on the plurality of training samples of the training dataset to provide a combined output; and

wherein, when updating the model weight of the second machine learning model based on the combined unnormalized output to provide the second trained machine learning model, the at least one processor is further programmed or configured to:

update the model weight of the second machine learning model based on the combined output to provide the second trained machine learning model.

7. A computer-implemented method for reducing dataset biases in natural language inference tasks using unadversarial training, the method comprising: performing, by at least one processor, an unadversarial training procedure to train a machine learning model to provide a trained machine learning model, wherein performing the unadversarial training procedure, comprises:

receiving a training dataset comprising a plurality of training samples, each training sample of the plurality of training samples includes transaction data associated with an electronic payment transaction;

generating a noise vector for the plurality of training samples based on a first uniform distribution;

perturbing each training sample of the plurality of training samples based on the noise vector to provide a plurality of perturbed training samples;

obtaining a gradient between a first output of the machine learning model that results from inputting each training sample of the plurality of training samples and a second output of the machine learning model that results from inputting each perturbed training sample of the plurality of perturbed training samples;

generating an updated noise vector based on the gradient; perturbing each training sample of the plurality of training samples based on the updated noise vector to provide a second plurality of perturbed training samples; and

updating a model weight of the machine learning model based on the second plurality of perturbed training samples to provide the trained machine learning model;

wherein the trained machine learning model is a first trained machine learning model that exploits bias features of the training dataset to make predictions, and wherein the method further comprises:

training a second machine learning model using a Product of Experts (POE) procedure based on the first trained machine learning model to provide improved predictions as compared to the first trained machine learning model.

8. The computer-implemented method of claim 7, wherein generating the noise vector for the plurality of training samples comprises:

initializing the noise vector with a uniform distribution, wherein initializing the noise vector comprises:

multiplying the first uniform distribution by a predefined radius to provide a second uniform distribution; and

dividing the second uniform distribution by a square root of an input embedding size of the machine learning model to provide an initial value of the noise vector.

9. The computer-implemented method of claim 7, wherein generating the updated noise vector comprises:

multiplying the gradient by a step size.

10. The computer-implemented method of claim 7, further comprising:

restricting a value of the updated noise vector based on a predefined radius to provide a second updated noise vector;

wherein perturbing each training sample of the plurality of training samples based on the updated noise vector to provide the second plurality of perturbed training samples comprises:

perturbing each training sample of the plurality of training samples based on the second updated noise vector to provide the second plurality of perturbed training samples.

11. The computer-implemented method of claim 9, wherein training the second machine learning model using the POE procedure comprises:

generating an unnormalized output from the first trained machine learning model based on the plurality of training samples of the training dataset;

generating an unnormalized output from the second machine learning model based on the plurality of training samples of the training dataset;

combining the unnormalized output from the first trained machine learning model and the unnormalized output from the second machine learning model to provide a combined unnormalized output; and

updating a model weight of the second machine learning model based on the combined unnormalized output to provide the second trained machine learning model.

12. The computer-implemented method of claim 11, wherein generating the unnormalized output from the first trained machine learning model based on the plurality of training samples of the training dataset comprises:

generating an output of the first trained machine learning model using a first logits function based on the plurality of training samples of the training dataset;

wherein generating the unnormalized output from the second machine learning model based on the plurality of training samples of the training dataset comprises:

generating an output of the second trained machine learning model using a second logits function based on the plurality of training samples of the training dataset;

wherein combining the unnormalized output from the first trained machine learning model and the unnormalized output from the second machine learning model to provide the combined unnormalized output comprises:

combining the output of the first trained machine learning model using the first logits function based on the plurality of training samples of the training dataset and the output of the second machine learning model using the second logits function based on the plurality of training samples of the training dataset to provide a combined output; and

wherein updating the model weight of the second machine learning model based on the combined unnormalized output to provide the second trained machine learning model comprises:

updating the model weight of the second machine learning model based on the combined output to provide the second trained machine learning model.

13. A computer program product for reducing dataset biases in natural language inference tasks using unadversarial training, the computer program product comprising at least one non-transitory computer-readable medium including one or more instructions that, when executed by at least one processor, cause the at least one processor to:

perform an unadversarial training procedure to train a machine learning model to provide a trained machine learning model, wherein, when performing the unadversarial training procedure, the one or more instructions cause the at least one processor to:

receive a training dataset comprising a plurality of training samples, each training sample of the plurality of training samples includes transaction data associated with an electronic payment transaction;

generate a noise vector for the plurality of training samples based on a first uniform distribution;

perturb each training sample of the plurality of training samples based on the noise vector to provide a plurality of perturbed training samples;

obtain a gradient between a first output of the machine learning model that results from inputting each training sample of the plurality of training samples and a second output of the machine learning model that results from inputting each perturbed training sample of the plurality of perturbed training

samples;

generate an updated noise vector based on the gradient;

perturb each training sample of the plurality of training samples based on the updated noise vector to provide a second plurality of perturbed training samples; and

update a model weight of the machine learning model based on the second plurality of perturbed training samples to provide the trained machine learning model;

wherein the trained machine learning model is a first trained machine learning model that exploits bias features of the training dataset to make predictions, and wherein the one or more instructions further cause the at least one processor to:

train a second machine learning model using a Product of Experts (POE) procedure based on the first trained machine learning model to provide improved predictions as compared to the first trained machine learning model.

14. The computer program product of claim 13, wherein, when generating the noise vector for the plurality of training samples, the one or more instructions cause the at least one processor to:

initialize the noise vector with a uniform distribution, wherein when initializing the noise vector with the uniform distribution, the one or more instructions cause the at least one processor to:

multiply the first uniform distribution by a predefined radius to provide a second uniform distribution; and

divide the second uniform distribution by a square root of an input embedding size of the machine learning model to provide an initial value of the noise vector.

15. The computer program product of claim 13, wherein, when generating the updated noise vector, the one or more instructions cause the at least one processor to:

multiply the gradient by a step size.

16. The computer program product of claim 13, wherein, the one or more instructions further cause the at least one processor to:

restrict a value of the updated noise vector based on a predefined radius to provide a second updated noise vector; and

wherein, when perturbing each training sample of the plurality of training samples based on the updated noise vector to provide the second plurality of perturbed training samples, the one or more instructions cause the at least one processor to:

perturb each training sample of the plurality of training samples based on the second updated noise vector to provide the second plurality of perturbed training samples.

17. The computer program product of claim 13, wherein, when training the second machine learning model using the POE procedure, the one or more instructions cause the at least one processor to:

generate an unnormalized output from the first trained machine learning model based on the plurality of training samples of the training dataset;

generate an unnormalized output from the second machine learning model based on the plurality of training samples of the training dataset;

combine the unnormalized output from the first trained machine learning model and the unnormalized output from the second machine learning model to provide a combined unnormalized output; and

update a model weight of the second machine learning model based on the combined unnormalized output to provide the second trained machine learning model.

18. The computer program product of claim 17, wherein, when generating the unnormalized output from the first trained machine learning model based on the plurality of training samples of the training dataset, the one or more instructions cause the at least one processor to:

generate an output of the first trained machine learning model using a first logits function based on the plurality of training samples of the training dataset;

wherein, when generating the unnormalized output from the second machine learning model based on the plurality of training samples of the training dataset, the one or more instructions cause the at least one to:

generate an output of the second trained machine learning model using a second logits function based on the plurality of training samples of the training dataset;

wherein, when combining the unnormalized output from the first trained machine learning model and the unnormalized output from the second machine learning model to provide the combined unnormalized output, the one or more instructions cause the at least one processor to:

combine the output of the first trained machine learning model using the first logits function based on the plurality of training samples of the training dataset and the output of the second machine learning model using the second logits function based on the plurality of training samples of the training dataset to provide a combined output; and

wherein, when updating the model weight of the second machine learning model based on the combined unnormalized output to provide the second trained machine learning model, the one or more instructions cause the at least one processor to:

update the model weight of the second machine learning model based on the combined output to provide the second trained machine learning model.