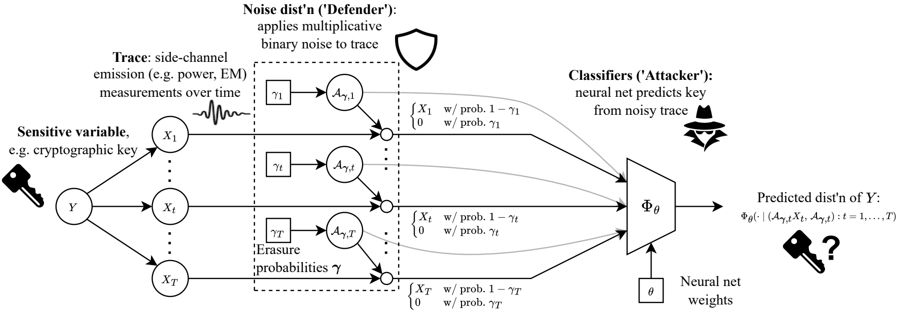

Figure 1: Schematic diagram of our Adversarial Leakage Localization (ALL) algorithm. Intuitively, ALL consists of a deep neural net $\Phi_{\boldsymbol{\theta}}$ which 'attacks' a cryptographic implementation by trying to predict sensitive variables Y (e.g. AES keys) from side-channel emission traces $\boldsymbol{X} := (X_1, \ldots, X_T)$. Simultaneously, a trainable multiplicative binary noise distribution 'defends' the implementation by distributing the noise $\boldsymbol{\mathcal{A}}_{\boldsymbol{\gamma}} \sim \prod_{t=1}^{T} \mathrm{Bern}(1-\gamma_t)$ to individual timesteps under a budget constraint to maximize the loss of the attacker. Because of the budget constraint, adding noise to one timestep *necessarily* entails removing it from other timesteps, and the 'defender' is forced to prioritize the timesteps with the highest utility to the 'attacker'. Thus, after training, we can learn the 'leakiness' of each timestep by the extent to which the 'defender' prioritized it.