Analisi di Sensibilità e Ottimizzazione dei Parametri

ChatGPT

September 3, 2024

Introduzione

$$P(rate_{fp} + rate_{tp} > th) \tag{1}$$

Introduzione

$$rate_{fp} = \mathcal{N}_{fp}(\mu_{fp}, \sigma_{fp}) \tag{2}$$

$$rate_{tp} = \mathcal{N}_{tp}(\mu_{tp}, \sigma_{tp}) \tag{3}$$

Introduzione

- FPA, False Positive Alert:

$$P(rate_{fp} + rate_{tp} \geq th) \cap P(rate_{tp} = 0) \tag{4}$$

Introduzione

- TPA, True Positive Alert:

$$P(rate_{fp} + rate_{tp} > th) \cap P(rate_{tp} > 0) \tag{5}$$

Simulazione

Si generano randomicamente:

- n false positives (fps).
- m true positives (tps), $m < n$.

adottando una distribuzione normale $\mathcal{N}(\mu, \sigma)$.

Simulazione /2

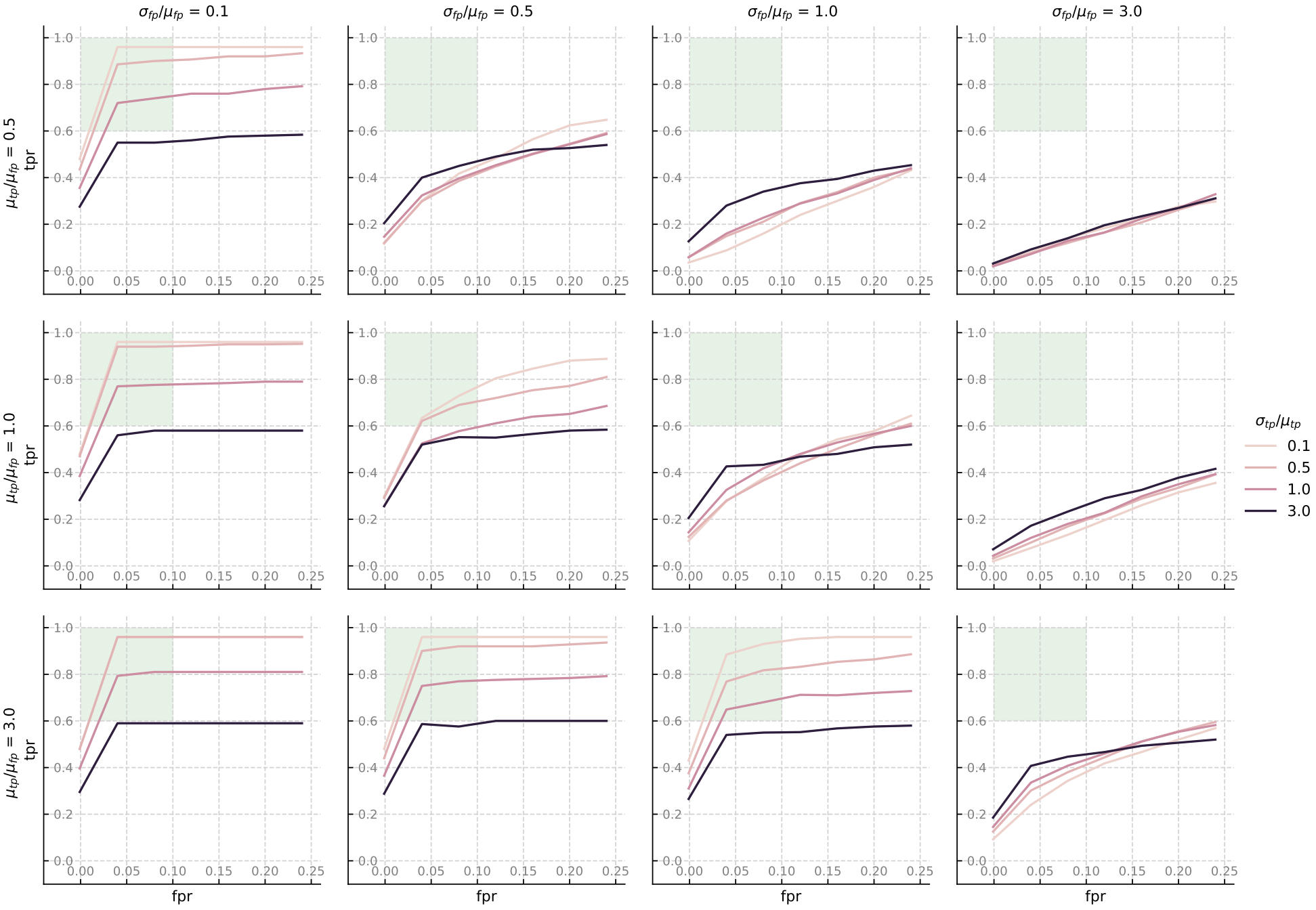
Si utilizzano i seguenti parametri:

- μ_{fp} viene scelta arbitrariamente e mantenuta fissa.
- σ_{fp} viene ottenuta dal rapporto σ_{fp}/μ_{fp}, il quale varia in un insieme di valori come, ad esempio, $\{0.1, 0.5, 1.0, 2.0\}$.
- μ_{tp} viene ottenuta dal rapporto μ_{tp}/μ_{fp}, il quale varia in un insieme di valori come, ad esempio, $\{0.5, 1.0, 3.0\}$.
- σ_{tp} si ottiene allo stesso modo di σ_{fp}.

parametro	calcolo	set
μ_{fp}	arbitrario e fisso	$\mu_{fp} \in 100$
μ_{tp}	$= \mu_{fp} * \mu_r$	$\mu_r = \mu_{tp}/\mu_{fp} \in \{0.5, 1, 3\}$
σ_{fp}	$= \mu_{fp} * r_{fp}$	$r_{fp} = \sigma_{fp}/\mu_{fp} \in \{0.1, 0.5, 1, 2\}$
σ_{tp}	$= \mu_{tp} * r_{tp}$	$r_{tp} = \sigma_{tp}/\mu_{tp} \in \{0.1, 0.5, 1, 2\}$

Simulazione /3

- Per ogni set di parametri, si calcola la ROC curve.
- La ROC curve analizza tutte le soglie disponibili *th* e mostra quali valori di FPR e TPR si ottengono per ciascuna di esse.
- Siccome nella NIDS il numero di falsi positivi deve essere minimo, la ROC curve che varia tra 0.0 e 1.0 è qui limitata a 0.0 e 0.25.


σ_{fp}/μ_{fp} = 0.1
σ_{fp}/μ_{fp} = 0.5
σ_{fp}/μ_{fp} = 1.0
σ_{fp}/μ_{fp} = 3.0
μ_{tp}/μ_{fp} = 0.5
μ_{tp}/μ_{fp} = 1.0
μ_{tp}/μ_{fp} = 3.0
tpr
fpr
0.0
0.2
0.4
0.6
0.8
1.0
0.00
0.05
0.10
0.15
0.20
0.25
σ_{tp}/μ_{tp}
0.1
0.5
1.0
3.0

Simulazione con *tps* veri

- Invece di generare i *tps*, utilizziamo le capture di Stratosphere.
- Suddividiamo la timeline di ogni capture in slot, finestre temporali adiacenti di lunghezza fissa.
- Per ogni slot:
  - Numero di predicted positive (*pp*), ovvero numero di domini che la rete LSTM ha predetto avere una probabilità > 0.5 di essere malevoli.

Simulazione con *tps* veri /2

- Si considerano *w* slots alla volta.
- Si generano con una distribuzione normale, *w* slots di *fps*.
- Si esegue questa operazione per tutta la timeline della capture malevola, scorrendo di uno slot per volta.
- Per ogni *overlap* si calcola la curva ROC.

Simulazione con *tps* veri /3

In questo caso i parametri precedenti saranno:

- μ_{tp_i} e σ_{tp_i}: calcolati dai dati reali della i-esima finestra.
- $\mu_{fp} = \mu_r \cdot \mu_{tp_i}$: quindi μ_{fp} viene ottenuto da μ_{tp_i}.
- σ_{fp} viene calcolato allo stesso modo di prima.

I parametri che possiamo variare sono quindi:

- μ_r, per ottenere μ_{fp} da μ_{tp}.
- r_{fp}, per ottenere σ_{fp} da μ_{fp}.

Simulazione con *tps* veri /3

Abbiamo scelto 4 capture per i seguenti malware:

- caphaw, simda, unknown, zbot.

, con i seguenti parametri:

- $\mu_r = \mu_{tp}/\mu_{fp} \in \{0.5, 1.0, 1.5\}$.
- $r_{fp} = \sigma_{fp}/\mu_{fp} \in \{0.1, 0.5, 1, 2, 4\}$.