SuTraN：ビジネスプロセスの全体文脈を考慮したサフィックス予測のためのエンコーダーデコーダートランスフォーマーモデル

著者：Brecht Wuyts, Seppe vanden Broucke, Jochen De Weerdt

翻訳者：本美桃佳

この論文では、ビジネスプロセスの予測モニタリング(PPM)におけるサフィックス予測を改善するため、SuTraN と呼ばれる新しいエンコーダーデコーダートランスフォーマーモデルを提案しています。従来のモデルが LSTM ベースで単一ステップの予測に依存するのに対し、SuTraN は全ての利用可能なデータを活用し、一回の計算で全てのイベントサフィックスを予測します。このモデルは特に、未来のイベントシーケンスをアクティビティラベル、タイムスタンプ、残り時間の観点から予測する性能に優れており、データ意識と自己回帰的な学習を活用しています。

実験の結果、SuTraN は現存する技術を全てのタスクで上回り、PPM における重要な要素であるデータ意識、シーケンス間学習、および残り時間予測を通じてその効果を証明しています。

SuTraN: ビジネスプロセスの全体文脈を考慮したサフィックス予測のためのエンコーダーデコーダートランスフォーマー

Brecht Wuyts[1], Seppe vanden Broucke[2] , Jochen De Weerdt[1]

（概要）プロセスマイニング（Process Mining, PM）における予測プロセスモニタリング（Predictive Process Monitoring, PPM）は、予測分析を用いてビジネスプロセスの進行を予測するものである。重要な課題の一つは「サフィックス予測」であり、アクティビティラベル、タイムスタンプ、残り実行時間を含む将来のイベントシーケンスを予測することである。現在の技術は、しばしば一歩先の予測に焦点を当てており、サフィックス生成には反復的なフィードバックループに依存し、ペイロードデータを十分に活用できていない。また、多くの技術がモデルアーキテクチャにおける最近の進展から遅れており、LSTM ベースのモデルに固執している状況である。

これらの課題に対処するため、本研究では PPM のサフィックス予測のための新しいトランスフォーマーアーキテクチャ「SuTraN」を提案する。SuTraN は反復的な予測ループを回避し、イベント特徴を含む利用可能なすべてのデータを活用して、一回のフォワードパスで全体のイベントサフィックスを予測するものである。このアプローチは、自己回帰的サフィックス生成、データ意識、シーケンス間学習（seq2seq）を統合している。実際のイベントログを用いた実験結果から、SuTraN がサフィックス予測において優れた性能を発揮することが示されている。これにより、既存の研究で見過ごされがちな重要な貢献が明らかとなった。

（キーワード）プロセスマイニング、予測プロセスモニタリング、ディープラーニング、トランスフォーマー、サフィックス予測、残り時間予測

Ⅰ. はじめに

予測プロセスモニタリングは、予測分析を活用し、将来のビジネスプロセスの軌道を予測することでプロセスマイニングを強化するものである。近年、ニューラルネットワークのような高度な機械学習モデルを含む進展により、特に構造化されていないプロセスにおいて、PPM の精度は大幅に向上している。このような進展により、PPM におけるディープラーニング（DL）ベースの技術が発展し、特にサフィックス予測のような複雑なタスクに焦点が当てられるようになっている。サフィックス予測とは、先行するイベント（プレフィックス）に基づき、将来のイベントシーケンス（サフィックス）を予測することであり、これにはアクティビティラベル、タイムスタンプ、残り実行時間の推定が含まれる。このような予測は、リソース計画、ボトルネック予測、期限管理、顧客サービス最適化など、さまざまなビジネスアプリケーションにおいて極めて重要である。

「シーケンス間学習」（sequence-to-sequence, seq2seq）ディープラーニングネットワークがイベントサフィックス生成に広く採用されているにもかかわらず、このパラダイムに基づいて訓練された技術は少なく、訓練手法と実際の展開の間にギャップが生じている。この制限、すなわち単一ステップ予測に焦点を当てること（文献[1]–[5]）は、全体のサフィックスを予測する際に不正確さを引き起こすだけでなく、動的なイベント特徴の利用を妨げ、文脈的な理解を制限している。現在、全体のイベントサフィックスを予測するために特化したモデルを提案している技術はわずか

[1] 情報システム工学研究センター（LIRIS），ルーヴェン・カトリック大学（KU Leuven），ベルギー・ルーヴェン

[2] ビジネス情報学およびオペレーション管理学部，ヘント大学，ベルギー

である（文献[6]–[8]）。さらに、多くのアプローチが、予測能力をさらに高める可能性のある追加のペイロードデータを見過ごしている（例：[1]–[5], [7], [8]）。

既存の多くのサフィックス予測技術、例えば文献[1]–[6], [8]においては、リカレントニューラルネットワーク（RNN）[10]の一種であるロング・ショートタームメモリ（LSTM）ネットワーク[9]に大きく依存している。一方、特にトランスフォーマーモデル[11]によるシーケンス間学習の最近の進展は、PPM においてほとんど未探究のままである（文献[7], [12], [13]）。

本論文では、PPM におけるイベントサフィックスおよび残り実行時間予測に特化した新しい全体文脈対応型エンコーダーデコーダートランスフォーマーネットワーク「Suffix Transformer Network（SuTraN）」を提案する。SuTraN は反復的な予測ループを排除し、単一のフォワードパスで全体のイベントサフィックスを予測する。シーケンス間学習、プレフィックスイベントの全ての利用可能な特徴の活用、自己回帰（AR）推論を組み合わせることで、SuTraN は進化するプロセスシーケンスに対する文脈的理解と適応性を強化するものである。

実際のイベントログを用いた広範な実験により、SuTraN が全ての予測タスクにおいて既存技術を上回る性能を示すことが確認されている。この評価は、SuTraN の高い性能を強調するだけでなく、AR 推論、データ意識、シーケンス間学習が PPM において正確な予測を実現するために重要であることを強調している。本研究では、これまで見過ごされがちであったこれらの側面に取り組むことで、効果的なサフィックス予測に必要な重要な要素について貴重な知見を提供するものである。

本論文の構成は以下の通りである。第 II 章では関連研究を概観し、第 III 章で SuTraN のアーキテクチャと手法を紹介する。第 IV 章では実験的評価を提示し、第 V 章で結果を議論する。最後に、第 VI 章で本研究の結論を述べる。

Ⅱ. 関連研究

近年の予測プロセスモニタリング（PPM）の進展により、ディープラーニング（DL）手法が、従来の機械学習（ML）やプロセスモデルに依存した技術を、サフィックス予測を含むさまざまな予測タスクにおいて上回ることが明らかになっている（文献[5], [6], [14], [15]）。文献分析により、DL ベースのサフィックス予測技術が、従来のプロセスモデルベースのアプローチから大きく分岐しつつあることが示されている。特に、表 I は既存技術の比較分析を提供し、シングルイベント予測（SEP）と完全な残りトレース予測（CRTP）の手法を区別している。この表の最後の列は、予測対象をアクティビティサフィックス（‘A’）、タイムスタンプサフィックス（‘T’）、残り時間（‘RT’）、その他（‘O’）として分類している。

SEP 技術（文献[1]–[5]）は、次のイベントを予測するためにシーケンスからベクトルへのモデルとして訓練される。当初はこの形で訓練されるが、推論段階では、反復的なフィードバックループを用いたシーケンス間学習（seq2seq）アプローチを採用し、各予測後にイベントプレフィックスを更新し、それらを新しいインスタンスとして次の反復に使用する。この自己回帰（AR）推論は、累積的な予測に基づいて各サフィックス予測を条件付けることで文脈的な関連性を強化するものの、動的なイベント特徴を予測しない限り、それらを活用できず、包括的な特徴利用を制限する。また、これらのモデルは、エンコーダーデコーダーアーキテクチャとしては分類されず[16]、エンコーダーが完全に自己回帰的に機能する単一のフォワードパスでサフィックスを生成する代わりに、外部の AR メカニズムに依存している。

表 I に挙げられているすべての SEP 技術は、それぞれ独自の特徴を持つ LSTM 層を使用している。たとえば、Evermann ら[3]は、アクティビティラベルのプレフィックスに基づいて次のアクティビティを予測する基本的な LSTM ネットワークを導入している。Lin ら[4]は、各予測に対して入力シーケンスの部分を異なる重み付けで処理する「モジュレータ」を追加してこれ

TABLE I
COMPARISON OF MOST RELEVANT APPROACHES FOR ACTIVITY AND TIMESTAMP SUFFIX PREDICTION.

	Approaches	Sequence-sequence training	AR Inference	Comprehensive Feature Utilization	Transformer (-based)	Encoder-Decoder	Prediction Targets
SEP	Evermann et al. [3]	×	✓	×	×	×	A
	Tax et al. [5]	×	✓	×	×	×	A / T / RT
	Lin et al. [4]	×	✓	×	×	×	A / O
	Di Francescomarino et al. [2]	×	✓	×	×	×	A
	Camargo et al. [1]	×	✓	×	×	×	A / T / RT / O
CRTP	Taymouri et al. [8]	✓	✓	×	×	✓	A / T / RT
	Ketykó et al. [7]	✓	✓	×	✓	✓	A / T / RT
	Gunnarsson et al. [6]	✓	×	✓	×	×	A / RT
	SuTraN (this paper)	✓	✓	✓	✓	✓	A / T / RT

を強化している。文献[2]の著者らは、推論時にプロセス実行の追加情報を取り入れることで予測を改善している。Tax ら[5]は、アクティビティラベルとタイムスタンプを同時に予測するための共有および専門化された LSTM レイヤーの組み合わせを探求し、さらに予測されたタイムスタンプから残り実行時間を予測している。Camargo ら[1]は、これに類似したアーキテクチャを採用し、訓練前に抽出された役割の予測も行っている。

対照的に、シーケンス間学習パラダイムに基づき、サフィックス予測のために特化して訓練された手法は少ない。たとえば、文献[6]は、LSTM ベースのモデルを用いて、一般的に使用されるタイムスタンプサフィックスアプローチとは異なるアクティビティ全体と残り実行時間サフィックスを予測している。この方法は、SuTraN と同様に、イベント特徴を含むすべてのプレフィックス特徴を活用しつつ、AR アプローチを使用せずにサフィックス全体を直接予測している。さらに、Taymouri ら[8]は、エンコーダーデコーダーLSTM アーキテクチャを導入し、デコーダーが AR 方式でアクティビティとタイムスタンプサフィックスを順次生成するものである。このアプローチは、文脈を動的に統合し、生成されたシーケンスの精度と関連性を向上させる。彼らは、教師付き学習を対敵的学習で補完し、ビームサーチにおけるビーム幅の拡大に伴う性能向上を示している。

一方で、Ketykó ら[7]は、サフィックス予測のためにトランスフォーマーエンコーダーデコーダーを含むさまざまなシーケンシャル DL アーキテクチャを比較している。[7]および[8]のどちらも、アクティビティラベルとタイムスタンプのみに焦点を当て、追加のペイロードデータを活用していない。実験的評価では、すべての予測タスクにおいてペイロードデータを完全に活用することで顕著な性能向上が示されており、PPM における包括的かつ文脈対応型サフィックス予測への有望なシフトを示している。

文献[7]で評価されたトランスフォーマーベースの技術を除き、トランスフォーマーコンポーネントを統合した手法は、次のイベント予測専用に提案されたものがわずかに存在するだけである（文献[12], [13], [17]）。これらはすべてエンコーダーのみのアーキテクチャを使用している。しかし、これらの方法は、SEP 技術が利用する反復的なフィードバックループを使用することで、サフィックス予測にも適用可能であると考えられる。

III. 方法論

A. 準備事項

1）イベントログデータ

プロセスマイニングにおけるイベントログ $L = \{\sigma i \mid 1 \leq i \leq |L|\}$は、ビジネスプロセスのケースやインスタンスを記録するものである（ここで $|L|$はケースの総数を表す）。各ケース σ_iは、時系列順に並べられたイベントのシーケンスとして表される $\langle ei, 1, \ldots, ei, n\rangle$（n は特定のケースにおいて実行されたイベントの数）である。簡単化のため、ケースを表す添字 iは省略される。イベント$e_j(\in \sigma)$を以下のように定義する：

定義 1（イベント）：

イベントは以下のようなタプルである：

$$e = \langle a, c, t, cf_1, \ldots, cf_{m_1}, ef_1, \ldots, ef_{m_2}\rangle$$

ここで、

- a: アクティビティラベル
- c: ケース ID
- t: タイムスタンプ
- $cf_1, \ldots, cf_{m_1}$: ケース特徴$(m_1 \geq 0)$
- $ef_1, \ldots, ef_{m_2}$: イベント特徴$(m_2 \geq 0)$

各イベントタプルの要素は個別にアクセス可能であり、添字で表される。たとえば、j-番目のイベント $(j \in 1, \ldots, n)$のアクティビティラベルは a_j、そのタイムスタンプはt_jである。
同じケースに属するすべてのイベント $e_j \in \sigma$ は同じケース ID$\left(\forall j, k \in \{1, \ldots, n\}: c_j = c_k\right)$および同じケース特徴$\left(\forall \alpha \in \{1, \ldots, m_1\}: cf_{\alpha,j} = cf_{\alpha,k}\right)$を共有する。一方で、イベント特徴は同じケースに属するイベント間で異なる。

2）サフィックス予測と残り時間予測

各ケース σ（完全なケース）から、プレフィックス-サフィックスペアの集合

$$\{(\sigma_p^k, \sigma_s^k) \mid 1 \leq k \leq n\}$$

を生成できる。ここで、

- プレフィックス $\sigma_p^k = \langle e_1, \ldots, e_k \rangle$は最初の kk 個のイベントを含む。
- サフィックス $\sigma_s^k = \langle e_{k+1}, \ldots, e_n \rangle$は最後の n-k 個のイベントを含む。

定義 2（サフィックス予測）：

与えられたプレフィックス σ_p^k に基づき、サフィックス σ_s^kに属するアクティビティとタイムスタンプのシーケンス、すなわち

$$\langle (a_{k+1}, t_{k+1}), \ldots, (a_n, t_n), (\text{EOS}) \rangle$$

を予測することを目的とする。ここで、EOS（End Of Sequence）トークンは、ケースの終了を示すために前処理で追加される。
さらに、残り時間予測は、最後に観測されたプレフィックスイベント e_k からケースの完了（e_n）までの総残り時間 $r_k = t_n - t_k$ を予測することを目的とする。

3）前処理

予測アルゴリズムがタイムスタンプおよびカテゴリ変数を解釈できるようにするため、これらを数値的な代理変数に変換する必要がある。タイムスタンプ情報は次の 2 つの数値特徴に変換される：

- $t_p^j = t_j - t_{j-1}$（前回のイベントからの経過時間、$\forall j = (2, \ldots, n)$
- $t_s^j = t_j - t_1$（ケース開始からの経過時間）

最初のイベント e_1では、これらの数値は 0 に設定される。
同様に、定義 2 のタイムスタンプサフィックスも

$$\langle t_p^{k+1}, \ldots, t_p^n, 0 \rangle$$

によって表される。カテゴリ特徴（アクティビティラベルを含む）は、ワンホットエンコードされたベクトルで数値的に表される。これらの疎なベクトルは、そのままモデルに供給される[5][8]か、事前学習済み[1]または学習可能[3][6][7][17]な線形射影（埋め込み）を使用してさらに処理される。本研究では後者を採用している（第 III-B 節参照）。

SuTraN に提示される最終的なプレフィックス表現（以降、「プレフィックスイベントトークンのシーケンス」と呼ぶ）は以下のように定義される：

定義 3（プレフィックスイベントトークンのシーケンス）:
各イベント $e_j \in \sigma_p^k$ が、以下を含むとする：

- mc_1 個のカテゴリケース特徴
- $mn_1 = m_1 - mc_1$個の数値ケース特徴
- mc_2 個のカテゴリイベント特徴
- $mn_2 = m_2 - mc_2$個の数値イベント特徴

前処理されたプレフィックスイベントトークンのシーケンスは以下で定義される：

$$\widetilde{\sigma_p^k} = \langle e_p^1, \ldots, e_p^k \rangle$$

ここで各プレフィックスイベントトークン $epj(\in \sigma \sim pk) e_p^j \left(\in \widetilde{\sigma_p^k}\right)$ は次のように表

される：

$$e_j^p = (a_j, t_j^p, t_j^s, (cf_{1,j}^c, \ldots, cf_{m_1^c,j}^c), (ef_{1,j}^c, \ldots, ef_{m_2^c,j}^c), (cf_{1,j}^n, \ldots, cf_{m_1^n,j}^n), (ef_{1,j}^n, \ldots, ef_{m_2^n,j}^n))$$

連続的な特徴（ターゲットを含む）は、正規分布($\sim N(0,1)$)へ標準化され、トレーニングセットの平均と標準偏差を使用してテストセットとバリデーションセットに適用される。数値特徴に欠損値がある場合、追加のバイナリ指標特徴（欠損値の場合は1、そうでない場合は0）を導入し、元の特徴には0を代入する。カテゴリ特徴の場合、欠損値には追加の「MISSING」カテゴリを採用し、トレーニングセットで見られなかったテストセットレベルには「OutOfVocabulary（OOV）」カテゴリを割り当てる。

B. SuTraN - Suffix Transformer Network

$$\pi(o_k^p) = \begin{cases} \langle \hat{a}_{k+1}, \ldots, \hat{a}_{k+D-1}, EOS \rangle & (1.\ activity\ suffix) \\ \langle \hat{t}_{k+1}^p, \ldots, \hat{t}_{k+D-1}^p \rangle & (2.\ timestamp\ suffix) \\ \hat{r}_k & (3.\ remaining\ time) \end{cases} \quad (1)$$

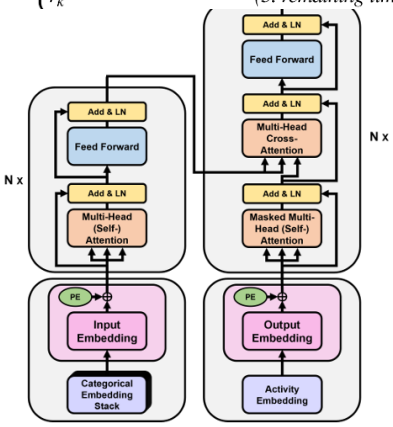

Fig. 1. *SuTraN* model architecture

SuTraN によって生成されたマルチモーダル（複数形式）の予測 $\pi(op_k)$（図1）は、プレフィックスイベントトークン $\widetilde{\sigma_p^k}$（定義3）が提示された際、次の式（式1）で表される。

$$\pi(o_k^p) = \begin{cases} \langle \hat{a}_{k+1}, \ldots, \hat{a}_{k+D-1}, EOS \rangle & (1.\ activity\ suffix) \\ \langle \hat{t}_{k+1}^p, \ldots, \hat{t}_{k+D-1}^p \rangle & (2.\ timestamp\ suffix) \\ \hat{r}_k & (3.\ remaining\ time) \end{cases} \quad (1)$$

SuTraN はエンコーダとデコーダの 2 つの主要な構成要素で構成されている。エンコーダは、各プレフィックスイベント $ej(\in \sigma pk) e_j\left(\in \sigma_p^k\right)$を処理してエンコードする。入力としてプレフィックスイベントトークンのシーケンス ~{σ_p^k} = ⟩⟨e_{p,1}, … , e_{p,k} を受け取り、同じ長さの dm-次元のプレフィックスイベント埋め込み $\langle h_{p,1}^N, \ldots, h_{p,k}^N \rangle$に変換する。これらの埋め込みは、各プレフィックスイベントの本質的な特徴を捉え、デコーダへの入力として機能する。

デコーダはこれらのプレフィックスイベント表現を利用して、逐次的（AR）に予測$\pi(op_k)$（式1）を生成する。より具体的には、各連続したデコードステップ $d = 1, \ldots, D$ において、エンコーダ埋め込み $\langle h_{p,1}^N, \ldots, h_{p,k}^N \rangle$と、現在までに生成された後続イベントトークンのシーケンス $\langle e_{s,0}, \ldots, e_{s,d-1} \rangle$に基づいて次の後続イベント $\pi_d(op_k)\left(\in \pi(op_k)\right)$ を予測する（式2）。各デコードステップの終了時に、新しい後続イベントトークン $e_{s,d} = \left(\widehat{a_{k+d}}, \widehat{t_{p,k+d}}, \widehat{t_{s,k+d}}\right)$が $\pi_d(op_k)$から派生される。

$$\pi_d(o_k^p) = \begin{cases} (\hat{a}_{k+d}, \hat{t}_{k+d}^p) & if\ d > 1 \\ (\hat{a}_{k+d}, \hat{t}_{k+d}^p, \hat{r}_k) & if\ d = 1 \end{cases} \quad (2)$$

最初のデコードステップ $d = 1$ のみ、追加のスカラーである残り時間予測 $\widehat{r_k}$が生成される。既存の手法 [1], [5], [7], [8] では、単に予測されたタイムスタンプ後続部分の合計を計算することで $\widehat{r_k} = \sum_{i=1}^{D-1} \widehat{t_{p,k+\iota}}$を導き出している。初期の実験では、タイムスタンプ予測の後続部分の合計に依存するよりも、追加の残り時間予測ヘッドを使用する方が、より正確な結果が得られることが示されている。タイムスタンプ予測そのものが誤差の影響を受けるため、この方法の方が効果的である。

このセットアップでは、最初に別個のエンコーダがプレフィックス全体を処理し、その後、各デコードステップでプレフィックス埋め込みと生成済みの後続部分の両方を考慮する AR デコーダが続く。このようにして、プレフィックスイベントに含まれる全ての追加的な特徴（動的なイベント特徴など）を利用しつつ、各連続した後続予測をそれまでの累積的な予測に直接条件付けすることが可能となる。この構造化されたアプローチにより、データに完全に基づいたプレフィックスイベントと、生成された後続部分という両方を考慮した、現実のケース進行に近い精緻で文脈に基づいた予測が可能となる。

次章では、エンコーダとデコーダの両方

についてさらに詳しく説明する。

1）SuTraN の構成要素

エンコーダとデコーダは非常に類似している。エンコーダは、初期プレフィックス埋め込みブロックと、N 個の同一のエンコーダブロックで構成される。同様に、初期後続埋め込みブロックと N 個の同一のデコーダブロックがデコーダを構成する。エンコーダブロックとデコーダブロックは [11] で紹介されたものと同一である。両ブロックの主要な要素は、マルチヘッドアテンション（MHA）層である。

MHA 層は、シーケンス $Hq(\in Rlq \times dm) = \langle hq,1,\dots,hq,lq\rangle H_q(\in R^{l_q\times dm}) = \langle h_{q,1},\dots,h_{q,l_q}\rangle$ 内の各ベクトル$(\in Rdm \in R^{dm})$を、シーケンス$Hk(\in Rlk \times dm) = \langle hk,1,\dots,hk,lk\rangle H_k(\in R^{l_k\times dm}) = \langle h_{k,1},\dots,h_{k,l_k}\rangle$の各ベクトルから情報を取り込むことで更新する。自己注意の場合、シーケンス H_q と H_k は同一である。

エンコーダブロック$l(\in \{1,\dots,N\})$は、前のブロック $l-1$ から受け取った k 個のプレフィックスイベント埋め込みをさらに更新し、更新されたシーケンス $\langle h\_\{p,1\}\^l,\dots,h\_\{p,k\}\^l\rangle$ を生成する。各ブロックは、マルチヘッド自己注意（MHSA）層と（位置ごとの）フィードフォワード（FF）層の 2 つのサブコンポーネントで構成される。MHSA サブ層は、同一シーケンス内の他の関連するベクトルから情報を取り込むことで、各プレフィックスイベント埋め込みを更新し、シーケンスの文脈や関係性に対する理解を深める。その後の FF 層は、各ベクトルを個別にさらに洗練し、データ内の複雑なパターンや関係性を捉えるための変換を適用する。

各サブ層には残差接続が配置され、その後に層正規化（Layer Normalization, LN）が続く。各サブ層は、$k \cdot dm$-次元のプレフィックスイベントベクトルのシーケンスを受け取り、さらに処理し、出力する。

同様に、デコーダブロック $l(\in \{1,\dots,N\})$は、前のブロックから受け取った後続イベント埋め込みのシーケンス $\langle h\_\{s,0\}\^\{l-1\},\dots,h\_\{s,d-1\}\^\{l-1\}\rangle$ をさらに処理し、更新された後続イベント埋め込み $\langle h\_\{s,0\}\^l,\dots,h\_\{s,d-1\}\^l\rangle$ を生成する。エンコーダブロックと比較して、デコーダブロックには追加のサブ層であるマルチヘッドクロスアテンション（MHCA）層が含まれている。この層はクロスアテンションを実行し、エンコーダによって生成された全てのプレフィックスイベント埋め込み $H_k = \langle h_\{p,1\}\^N,\dots,h_\{p,k\}\^N\rangle$から取得した関連情報で各後続イベント埋め込みをさらに更新し、イベントプレフィックスからのグローバルな文脈を生成プロセスに組み込む。

さらに、デコーダの自己注意層（Masked Multi-Head Self-Attention Layer）は、未来のイベント情報にアクセスすることを防ぎ、後続部分の生成において AR プロパティを維持する点で、エンコーダの注意層とは異なる。

エンコーダとデコーダブロックがプレフィックスおよび後続イベント埋め込みをさらに処理できるようにするため、プレフィックスイベントトークン $\langle e\_\{p,1\},\dots,e\_\{p,k\}\rangle$ と後続イベントトークン $\langle e_\{s,0\},\dots,e_\{s,d-1\}\rangle$は、それぞれが動作する dm-次元の潜在空間に射影される必要がある。この変換は、それぞれ初期プレフィックス埋め込みブロックおよび初期後続埋め込みブロックによって実行される。

IV. 実験設定

実験は Python 3.10、PyTorch 2.0、CUDA 11.0 を用いて実施され、NVIDIA Quadro RTX 5000 GPU（16GB RAM）上で実行された。SuTraN は、エンコーダおよびデコーダブロックを $N = 4$、埋め込み次元を$d_m = 32$ に設定して実装された。この設定は、初期実験においてモデル性能と計算複雑性の間の良好なトレードオフであることが示された。全てのマルチヘッドアテンション層は $M = 8$ 個の並列アテンションヘッドを含み、それぞれのヘッドの次元は 4（$\frac{d_m}{M} = \frac{32}{8} = 4$）であった。フィードフォワード層（FF 層）はすべて、次元 $d_{ff} = 128$ の隠れサブ層と ReLU 活性化関数を含んでいた。

SuTraN は最大 200 エポックで学習さ

れ、AdamW オプティマイザ（減衰率 0.00010.0001 と初期学習率 0.00020.0002）を用い、指数的な学習率スケジューラ（減衰因子 0.960.96）で学習率が調整された。検証スコアが 3 つの予測ターゲット全体で 24 エポック連続して改善しない場合、早期終了が適用された。最終的な重みは、3 つの予測タスク全体で検証性能が最良のトレードオフを提供したエポックに基づいて選択された。

アクティビティ後続部分とタイムスタンプ後続部分の予測において、学習プロセスを効率化し、モデルの基本機能を強調するために教師強制が使用された。さらに、マルチタスク学習アプローチとして損失関数の最適化が同時に行われた。それぞれのターゲットに対して以下の損失関数が計算された。

1) アクティビティ後続部分予測 - カテゴリカルクロスエントロピー

$$\mathscr{L}_{\text{activity}} = -\frac{1}{\sum_{i=1}^{B} N_i} \sum_{i=1}^{B} \sum_{t=1}^{N_i} \sum_{c=1}^{C} a_{i,t,c} \log(\hat{a}_{i,t,c}) \tag{4}$$

- B (= 128): バッチサイズ
- N_i: i番目のインスタンスの（真実値）後続部分に含まれるイベント数
- C: アクティビティラベルの総数
- $a_{i,t,c}$: i番目のインスタンスの tt 番目のイベントにおける真のアクティビティラベル
- $\widehat{a_{i,t,c}}$: 予測されたアクティビティラベルの確率

2) タイムスタンプ後続部分予測 - 平均絶対誤差（MAE）

$$\mathscr{L}_{\text{timestamp}} = \frac{1}{\sum_{i=1}^{B} N_i} \sum_{i=1}^{B} \sum_{t=1}^{N_i} |\hat{t}_{i,t} - t_{i,t}| \tag{5}$$

- $\widehat{t_{i,t}}$: i番目のインスタンスの tt 番目のイベントにおける予測タイムスタンプ
- $t_{i,t}$: 真のタイムスタンプ

3) 残り実行時間予測 - 平均絶対誤差（MAE）

$$\mathscr{L}_{\text{runtime}} = \frac{1}{B} \sum_{i=1}^{B} |\hat{r}_i - r_i| \tag{6}$$

- $\hat{r}_i$: i番目のインスタンスの予測残り実行時間
- r_i: 真の残り実行時間

最終的に、マルチタスク学習で使用される損失関数は以下の未加重和として定義される。

$$\begin{aligned} Lbatch = Lactivity + Ltimestamp \\ + LruntimeL_{\text{batch}} \\ = L_{\text{activity}} + L_{\text{timestamp}} \\ + L_{\text{runtime}} \end{aligned}$$

残り時間およびタイムスタンプ後続部分のラベルを標準化することで、MAE 損失スケールをカテゴリカルクロスエントロピー損失とよりよく整合させた。また、タイムスタンプおよび残り時間のイベントログにおける大きな外れ値に対してモデルのロバスト性を向上させるため、平均二乗誤差（MSE）ではなく MAE が選択された。

A. データ

SuTraN は、4 つの実データイベントログを用いて複数のベンチマーク手法と比較して評価された。そのうち 3 つは公開されており（BPIC17、BPIC17-DR、BPIC19）、それらの主な特性は本節で説明する前処理を基に表 II に要約されている。特に、BPIC17 のイベントログには、同じアクティビティの即時繰り返しが頻繁に見られる。この問題を緩和するために、元のプロセスとケースを維持しつつ、同じアクティビティの連続した繰り返しを除去した BPIC17-DR（"Duplicates Removed"）バージョンを作成した。この前処理により、イベント数が 40 万件以上削減され、ユニークなアクティビティシーケンス数（制御フローのバリエーション）が 14,745 から 3,592 に減少した。BPIC17-DR を簡略化することで、予測モデル評価の明確性と信頼性が向上し、BPIC17 の複雑性が任意の繰り返しにより人工的に増大し、データにノイズを導入していた可能性が示唆された。

BAC イベントログは、ヨーロッパの大規模空港の手荷物処理システムから取得されたが、公開はされていない。

各ケース $\sigma = \langle e\_1, \ldots, e\_n \rangle$ は、n 個のプレフィックスイベントトークンシーケンス$\widetilde{\sigma_p^k}(nk \in \{1, \ldots, n\})$に分割された（定義 3 参照）。各プレフィックスには、アクティビティ後続部分、タイムスタンプ後続部分

（前のイベントからの経過時間で表される）、およびスカラーの残り時間ラベルが関連付けられている。SuTraN（および IV-B 節で議論される ED-LSTM ベンチマーク）に教師強制を適用するために、各プレフィックスに対応する後続イベントトークンシーケンス（III-B 節参照）も導出された。

データリークを防ぐため、[18] のガイドラインに従い、75-25% の時間外トレイン-テスト分割方法を採用した。得られたトレーニングセットは、ケースの最後の 20% を検証セットに割り当てることで、最終トレーニングセットと検証セットにさらに分割された。BPIC19 データセットでは、2018 年 9 月以降にスループット時間の平均が急激に低下する異常 [19] が観察されたため、修正された分割手順が必要であった。この異常は、イベントログの抽出方法に起因すると考えられる。この調整は、テストセットのバイアスを軽減することを目的としている。

プレフィックス生成の前に、イベント数が最も多いケースの上位 1.5% を外れ値として除外し、その過大な影響を緩和した。これらの外れ値を含めると、トレーニングに必要なシーケンス長が過度に大きくなり、計算コストとモデル学習の複雑性が増加するためである。数値的特徴とターゲットは、ケースではなくトレーニングセットプレフィックスから計算された平均と標準偏差を使用して標準化され、各ケース内の複数の異なるプレフィックス間で一貫したスケーリングが保証された。

B. ベンチマーク手法

[7], [18] によると、前処理と評価設定の違いが原因で、研究間での結果の直接比較が複雑化または困難になることがある。SuTraN の性能を既存手法と公平かつ統制された条件下で評価し、特に以下の要素を比較するために、既存手法を再実装し、全てのベンチマークでデータの前処理とスケーリングを標準化した。

1. **モデリング設定**: 1 ステップ先の予測（SEP） vs. 後続部分全体の予測（CRTP）
2. **ネットワーク構成**: LSTM vs. Transformer
3. **文脈の考慮**: データ対応（DA） vs. 非データ対応（NDA）

全ての DA 手法は、定義 3 で定義されたプレフィックスイベント表現を使用する。一方で、NDA 手法ではプレフィックスイベントトークンはアクティビティラベルと 2 つの時間的特徴(a_j, t_p, t_s)に制限される。SuTraN と同様に、カテゴリカル変数は学習済みの埋め込みを用いて処理され、数値的特徴と結合された。

役割予測（[1]）のような特定の手法固有のターゲットは除外され、アーキテクチャおよびモデリングパラダイムの違いによる性能への影響のみを検証した。この統制された実験設計により、外部的な性能干渉を最小化し、調査対象となる要因の正確な評価を可能にした。

いくつかの既存アーキテクチャは、タスクの類似性、コードの可用性、実装の容易性、および競争力のある性能に基づいて適応され、再実装された。表 III は、それぞれの特性と共に全ての実装を示している。

SEP-LSTM 手法については、[5] によって紹介されたアーキテクチャを採用した。この手法は、[1] のマルチタスクモデルと類似しているが、後者で使用された追加の役割予測 LSTM 層は省略されている。[5] で報告された最適なパラメータ設定を使用してトレーニングを行った。

C. 評価

全ての AR モデル（SuTraN、ED-LSTM、SEP-LSTM）は、アクティビティおよびタイムスタンプ後続部分を逐次生成した。各デコードステップで、予測されたアクティビティラベルとタイムスタンプが後続イベントトークンを更新するために使用された。SEP-LSTM の場合、新しいプレフィックスイベントトークンを作成するために使用された。

V. 結果

表 IV は、4 つのイベントログにおける SuTraN の性能を示している。最良の結果は太字かつ下線付きで、次点の結果は太字

のみで強調されている。SuTraN は、残り時間予測において一貫してベンチマークを上回り、アクティビティ後続部分およびタイムスタンプ後続部分の予測においても、4 つのイベントログ中 3 つで優れた性能を発揮した。ただし、BPIC17 においては CRTP-LSTM がより高い DLS スコアを達成し、ED-LSTM がタイムスタンプ後続部分の予測で SuTraN およびその NDA バージョンをわずかに上回った。

データ対応（DA）はモデル性能を大幅に向上させる。DA モデル（SuTraN と CRTP-LSTM）は、アクティビティ後続部分および残り時間予測において、一貫して非データ対応（NDA）モデルを上回った。この利点は、タイムスタンプ後続部分の予測でも確認されたが、2 つの軽微な例外を除く。

SuTraN の逐次生成（AR）アプローチは、アクティビティ後続部分の予測において CRTP-LSTM より優れており、BPIC17 を除く 4 つのイベントログ中 3 つでこの傾向が見られた。同様の傾向が、より控えめながら、2 つのイベントログにおける NDA バージョン間の比較でも観察された。BPIC19 イベントログでは同点となった。

さらに、SuTraN は、残り時間、アクティビティ後続部分、タイムスタンプ後続部分予測のすべてに対して明示的に学習を行う唯一のモデルであり（表 III）、この包括的なマルチタスク学習アプローチが有利であることが証明された。これにより、全ログで残り時間予測が優れた性能を示し、タイムスタンプ後続部分の予測では 1 つを除く全ログで優れた性能を発揮した。

CRTP 手法（SuTraN、ED-LSTM、CRTP-LSTM）の優位性は、SEP 手法（SEP-LSTM）の結果と比較することで明確に示されている。SEP-LSTM は通常、最も低い順位を 5 回記録し、次いで CRTP-LSTM（NDA）が 4 回最下位となった。この結果は、DA の後続部分予測と AR の後続部分生成の重要性を強調している。特に、CRTP-LSTM を非データ対応からデータ対応に変更すると、性能が大幅に向上し、一般的に 2 番目に悪いモデルから 2 番目に優れたモデルへと変化した。

AR 後続部分生成モデル（SuTraN およびその NDA バージョン、ED-LSTM）は、非 AR モデルよりも優れた性能を発揮した。SuTraN 実装は最下位の区分を完全に回避し、ED-LSTM は 2 回のみ最下位となった。これは、ED-LSTM が残り時間をタイムスタンプ予測から導出する際に誤差が増加することに起因している。

SuTraN（NDA）と ED-LSTM を比較すると、PPM（プロセスマイニング）の後続部分生成におけるトランスフォーマーの可能性が強調される。パラメータ数が半分以下であるにもかかわらず、SuTraN（NDA）はアクティビティ後続部分およびタイムスタンプ後続部分の予測で ED-LSTM とよく競合し、残り時間予測では ED-LSTM を上回った（表 III）。

[7] で指摘されたように、イベントログのトレース長が右に偏っているため、平均的なメトリクスだけでは DL アーキテクチャを PPM で比較するのに十分ではない可能性がある。このため、図 2 は、プレフィックス長および後続部分長の関数としてアクティビティ後続部分（DLS）および残り時間（MAE）の予測メトリクスを示している。プロット上の各点は、特定のプレフィックスまたは後続部分の長さを持つテストインスタンスの平均 DLS または MAE を表し、モデルの性能が観測されたイベント数および予測される後続部分のイベント数に応じてどのように変化するかについての洞察を提供している。右側の縦軸は、それぞれの長さに該当するインスタンス数を示している。

直感的には、プレフィックスが長いほど性能が向上し、後続部分が短いほど性能が低下することが期待される。しかし、[7] で観察されたように、この傾向は特にプレフィックスが長い場合には一貫していない。これらの拡張されたケースには、明確な構造がない繰り返しアクティビティが含まれることが多く、モデリングにおける代表性について疑問を投げかけ、モデルがそのような異常に適応するべきか、それとも耐性を保つべきかという問題を提示している。

TABLE IV
PERFORMANCE COMPARISON ACROSS DIFFERENT TECHNIQUES AND DATASETS.

	Damerau-Levenshtein Similarity (DLS)				MAE Timestamp Suffix Prediction				MAE Remaining Runtime Prediction			
	BPIC17-DR	BPIC17	BPIC19	BAC	BPIC17-DR	BPIC17	BPIC19	BAC (sec.)	BPIC17-DR	BPIC17	BPIC19	BAC (sec.)
SEP-LSTM	0.6733	0.2160	0.8425	0.7206	**1178**	762	16604	113	10139	11823	30572	420
CRTP-LSTM (NDA)	0.6525	0.3357	0.8435	0.7320	1391	1009	17182	113	8931	8906	29323	318
CRTP-LSTM	**0.6741**	**0.4095**	**0.8522**	**0.8374**	1556	996	**15708**	112	**8000**	**8685**	**21345**	**301**
ED-LSTM	0.6737	0.3239	0.8477	0.7424	1200	**739**	16485	**108**	9705	12160	31000	338
SuTraN (NDA)	0.6723	0.2669	0.8435	0.7355	1201	**745**	16621	109	8896	8860	29209	308
SuTraN	**0.7274**	**0.3843**	**0.8699**	**0.8461**	**1157**	749	**14542**	**106**	**7727**	**7913**	**20182**	**290**

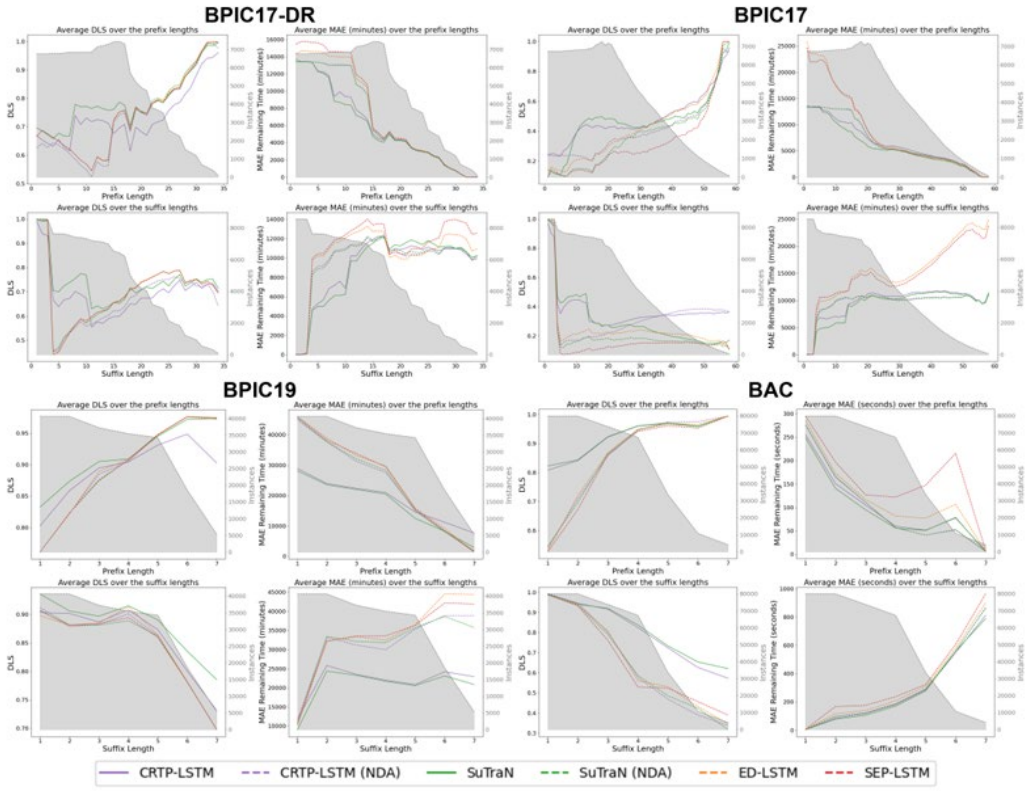

Fig. 2. Performance metrics in function of prefix and suffix lengths.

さらに、これらの外れ値は、トレーニングおよびテストセットの両方で顕著に過小評価されており、特に最長のプレフィックスおよび後続部分について性能評価を歪めている。これらの外れ値ケースから派生したインスタンスは、全長にわたる平均メトリクスに対して不均衡な影響を及ぼす一方、より典型的なケース長は、それぞれの長さまでの測定値にのみ影響を及ぼす。

これらのケース長分布が右に偏っている中での低い表現度を考慮し、こうした課題を軽減し、異なるプレフィックスおよびサフィックス長にわたるモデル性能の代表的な評価を保証するために、ケース長の第 5 パーセンタイルから第 90 パーセンタイル内にあるケースから導出されたテストインスタンスに焦点を当てた。

図 2 の分析では、いくつかの興味深いパターンが明らかになった。DA（データ対応）モデルは、初期段階（短いプレフィックス長）から一貫してアクティビティ後続部分の予測で優れており、NDA（非データ対応）モデルは通常、より多くのイベントが展開される（長いプレフィックス長）につれて追いつく。この傾向はすべてのイベントログで一貫して見られる。一方で、NDA モデルは短いサフィックスでは競争力があるものの、サフィックス長が増加するにつれて性能が急速に低下する。このことは、DA モデルが長いシーケンス全体でより高品質なアクティビティ後続部分を生成することを示している。

BPIC17-DR ログの DLS（Damerau-Levenshtein Similarity）サフィックスプロ

ットでは、サフィックス長の増加に伴いDLS スコアが初めは増加し、その後減少するという注目すべきパターンが現れる。これに対し、BPIC17 ログでは主に CRTP-LSTM の DLS スコアがサフィックス長の増加とともに上昇しており、表 IV で報告された高い DLS スコアを説明する要因となっている可能性がある。BPIC17 の長いケースには、同じアクティビティの連続した繰り返しが多数含まれることが多い。CRTP-LSTM の非 AR（逐次生成しない）アクティビティ後続部分生成は、これらの繰り返しに対して過剰適合し、生成中にそれに固執する可能性がある。この現象はDLS メトリクスによってノイズの多い後続部分に対して不幸にも高い評価を与える。

残り時間予測の MAE（平均絶対誤差）の変化にも類似のパターンが見られる。DA モデルは短いプレフィックスに対して一貫して低い残り時間誤差を達成し、包括的なデータ入力を活用している。一方で、NDA モデルはサフィックス長が増加するにつれて予測精度の低下が急速に進む。この効果は、平均値に対するスループット時間の変動が大きい BAC イベントログでは最も顕著でない。この変動には、データに含まれていない要因（例: フライトの遅延、ストライキ、休日など）が寄与している可能性がある。

さらに、明示的な残り時間予測は特に有利であることが、BPIC17 および BPIC19 イベントログのサフィックス長に対する MAE プロットで示されている。SuTraN や CRTP-LSTM、およびそれらの NDA 対応バージョンは、長いサフィックス長に対しても SEP-LSTM や ED-LSTM より低い MAE 値を報告している。後者の 2 つの手法では、残り時間の予測がタイムスタンプ後続部分の予測の合計に依存しており（これ自体が予測誤差を含む）、さらに予測されたアクティビティ後続部分における EOS（シーケンス終了）の正確性にも依存するため、EOS が早すぎるまたは遅すぎる可能性がある。アクティビティ後続部分の予測品質がサフィックス長の増加とともに低下するため、この影響は残り時間予測をさらに悪化させる。こうした影響は、長いケースで反復的でノイズの多いアクティビティパターンを特徴とする BPIC17 イベントログで最も顕著である。

これに対し、SuTraN、CRTP-LSTM、およびそれらの NDA 対応バージョンは、さまざまなサフィックス長にわたって比較的安定した MAE 値を維持している。

VI. 議論と結論

本研究では、プロセスマイニング（PPM）におけるマルチタスク後続部分予測のために設計された、完全な文脈認識型エンコーダ-デコーダトランスフォーマーネットワークである SuTraN を紹介した。SuTraN は、従来技術の限界に対処し、単一のフォワードパスでイベント後続部分全体を予測する点で独自の特徴を持つ。具体的には、seq2seq（シーケンス間学習）、逐次生成（AR 後続部分生成）、明示的な残り時間予測、包括的なデータ対応を統合している。

SuTraN の性能を厳密に評価するために、既存技術をゼロから再実装し、すべてのモデルでデータの前処理およびスケーリングを統一した。この標準化されたアプローチにより、既存文献との徹底的な比較が可能となり、SuTraN の統合機能がもたらす性能向上が明らかになった。結果として、SuTraN は残り時間、アクティビティ後続部分、タイムスタンプ後続部分を同時に予測する上での優位性を示した。

本研究の結果は、以下の要素が PPM タスクにおける優れた性能の鍵となることを強調している。

- AR デコード
- データ対応
- seq2seq 学習
- 明示的な残り時間予測

これらの要素は従来技術ではしばしば見過ごされていたが、SuTraN にはユニークに統合されている。SuTraN はこれらの全ての次元で卓越しており、すべてのタスクにおいてベンチマークを上回り、その有効性と堅牢性を示している。

また、プレフィックスおよびサフィックス長の変動に伴う性能の分析から、DA（データ対応）モデルは、初期プロセス段階（短いプレフィックス長）および長いサ

フィックスの生成が必要なインスタンスで、著しく高い精度を発揮することが明らかになった。特に明示的な残り時間予測は、長いサフィックス長や複雑な制御フローを持つイベントログが関与するシナリオにおいて有利であることが証明された。一方で、暗黙的な予測手法は、精度が低いタイムスタンプ後続部分予測に依存しているため、これらのシナリオでは精度を維持できない。

結果は、PPM におけるトランスフォーマーの可能性と適合性を示唆しているものの、トランスフォーマーと LSTM に基づくモデルの優越性に関する決定的な結論を得るには、さらなる研究が必要である。将来的な研究では、異なるログやその固有の特性にわたってこれらのアーキテクチャを包括的に比較するために設計された統制された実験を通じて、この問題をさらに探求できるだろう。

研究コミュニティにおける透明性と協力を促進するために、本研究のコード（すべての再実装を含む）を以下の URL で公開している：
https://github.com/BrechtWts/SuffixTransformerNetwork
リポジトリには、詳細なドキュメント、前処理手順の包括的な説明、すべての再実装の詳細な記述、および PPM 分野の発展を促進し再現性を確保するための補足資料が含まれている。

参考文献

[1] M. Camargo, M. Dumas, and O. González-Rojas, “Learning accurate lstm models of business processes,” in Business Process Management, T. Hildebrandt, B. F. van Dongen, M. Röglinger, and J. Mendling, Eds. Cham: Springer International Publishing, 2019, pp. 286–302. [2] C.DiFrancescomarino, C. Ghidini, F. M. Maggi, G. Petrucci, and A. Yeshchenko, “An eye into the future: Leveraging a-priori knowledge in predictive business process monitoring,” in Business Process Management, J. Carmona, G. Engels, andA.Kumar,Eds. Cham:SpringerInternationalPublishing, 2017,pp. 252–268. [3] J. Evermann, J.-R. Rehse, and P. Fettke, “Predicting process behaviour using deep learning,” Decision Support Systems, vol. 100, pp. 129–140, 2017, smart Business Process Management. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167923617300635 [4] L. Lin, L. Wen, and J. Wang, “Mm-pred: A deep predictive model for multi-attribute event sequence,” in Proceedings of the 2019 SIAM international conference on data mining. SIAM, 2019, pp. 118–126. [5] N. Tax, I. Verenich, M. La Rosa, and M. Dumas, “Predictive business process monitoring with LSTM neural networks,” in Proceedings of the 29th International Conference on Advanced Information Systems Engineering. Springer, 2017, pp. 477–492. [6] B. R. Gunnarsson, S. v. Broucke, and J. De Weerdt, “A direct data aware lstm neural network architecture for complete remaining trace and runtime prediction,” IEEE Transactions on Services Computing, vol. 16, no. 4, pp. 2330–2342, 2023. [7] I. Ketykó, F. Mannhardt, M. Hassani, and B. F. van Dongen, “What averages do not tell: predicting real life processes with sequential deep learning,” in Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing, ser. SAC’22. NewYork,NY,USA:Association for Computing Machinery, 2022, p. 1128–1131. [Online]. Available: https://doi.org/10.1145/3477314.3507179 [8] F. Taymouri, M. L. Rosa, and S. M. Erfani, “A deep adversarial model for suffix and remaining time prediction of event sequences,” in Proceedings of the 2021 SIAM International Conference on Data Mining, SDM 2021, Virtual Event, April 29- May 1, 2021, C. Demeniconi and I. Davidson, Eds. SIAM, 2021, pp. 522–530. [Online]. Available: https://doi.org/10.1137/1.9781611976700.59 [9] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997. [10] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” Nature, vol. 323, no. 6088, pp. 533–536, Oct 1986. [Online]. Available: https://doi.org/10.1038/323533a0 [11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is all you need,” in Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf [12] Z. A. Bukhsh, A. Saeed, and R. M. Dijkman, “Processtransformer: Predictive business process monitoring with transformer network,” 2021. [13] P. Philipp, R. Jacob, S. Robert, and J. Beyerer, “Predictive analysis of business processes using neural

networks with attention mechanism," in 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), 2020, pp. 225–230. [14] W. Kratsch, J. Manderscheid, M. Röglinger, and J. Seyfried, "Machine learning in business process monitoring: A comparison of deep learning and classical approaches used for outcome prediction," Business & Information Systems Engineering, vol. 63, no. 3, pp. 261–276, Jun 2021. [Online]. Available: https://doi.org/10.1007/s12599-020-00645-0 [15] I. Verenich, M. Dumas, M. L. Rosa, F. M. Maggi, and I. Teinemaa, "Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring," ACM Trans. Intell. Syst. Technol., vol. 10, no. 4, jul 2019. [Online]. Available: https://doi.org/10.1145/3331449 [16] I.Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in Advances in Neural Information Processing Systems, Z. Ghahra mani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf [17] J. Moon, G. Park, and J. Jeong, "Pop-on: Prediction of process using one-way language model based on nlp approach," Applied Sciences, vol. 11, no. 2, 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/2/864 [18] H. Weytjens and J. De Weerdt, "Creating unbiased public benchmark datasets with data leakage prevention for predictive process monitoring," in Business Process Management Workshops, A. Marrella and B. Weber, Eds. Cham: Springer International Publishing, 2022, pp. 18–29. [19] B. Wuyts, H. Weytjens, S. vanden Broucke, and J. De Weerdt, "DyLoPro: Profiling the dynamics of event logs," in Business Process Management, C. Di Francescomarino, A. Burattin, C. Janiesch, and S. Sadiq, Eds. Cham: Springer Nature Switzerland, 2023, pp. 146–162.